Vysoká škola báňská – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Řízení síťového provozu v bezdrátových sítích

Network traffic control in Wireless Networks

2019 **Bc. David Krmela**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání diplomové práce

Student:	**Bc. David Krmela**
Studijní program:	N2647 Informační a komunikační technologie
Studijní obor:	2601T013 Telekomunikační technika
Téma:	Řízení síťového provozu v bezdrátových sítích Network Traffic Control in Wireless Networks
Jazyk vypracování:	čeština

Zásady pro vypracování:

Při bezdrátové komunikaci je potřebné řešit propustnost sítě. Cílem diplomové práce je navrhnout řešení pro optimální datové přenosy s využitím řízení provozu.

Řešení práce spočívá ve splnění následujících bodů:
1. Seznámení a popis základních parametrů bezdrátových technologií.
2. Seznámení a popis parametrů QoS.
3. Návrh řešení pro řízení síťového provozu.
4. Testování v provozních podmínkách.
5. Zhodnocení a analýza dosažených výsledků.

Seznam doporučené odborné literatury:

[1]Giambene, G. *Queuing Theory and Telecommunications: Networks and Applications.* Springer 2014, ISBN-13: 978-1461440833

[2]Chee-Hock Ng, Soong Boon-Hee *Queueing Modelling Fundamentals: With Applications in Communication Networks.* Wiley 2008, ISBN-13: 978-0470519578

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Pavel Nevlud**

Datum zadání: 01.09.2016
Datum odevzdání: 30.04.2019

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta elektrotechniky a informatiky 1D16 OSTRAVA

prof. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne: *30. dubna 2019*

.....
podpis studenta

Poděkování

Rád bych poděkoval panu Ing. Pavlu Nevludovi, za odbornou pomoc a konzultaci při vytváření této diplomové práce.

Prohlášení zástupce spolupracující právnické nebo fyzické osoby

„Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v diplomových programech VŠB-TU Ostrava.“

Dne: *30. dubna 2019*

……..……
podpis zástupce

Abstrakt

Diplomová práce se zabývá návrhem a realizací bezdrátové sítě a serveru, který řídí datové toky do interní sítě a zpátky do venkovní sítě. Dále konfigurací serveru a tvorbě statické směrovací tabulky. V bezdrátové síti se dále nachází směrovací prvky na bázi UNIX, které jsou umístěny ve vnitřní bezdrátové síti. Tyto zařízení mají opět staticky nastavené směrovací tabulky. Zařízení ve vnitřní síti jsou kombinovanou formou ethernetu a bezdrátového přenosu.

Základní požadavek je rozdělení určitého garantované přenosové rychlosti mezi služby ve vnitřní síti. Přidělení garantované přenosové rychlosti probíhá připojením síťového kabelu na síťovou kartu serveru. Server dále řídí síťový provoz a třídí jej podle priorit do tříd. Dále se v síti využívá na všech lokálních směrovačích řízení tvarování síťového provozu.

Klíčová slova

Algoritmus, bezdrátová síť, CBQ, cyklus, ethernet, filtr, HTB, IP adresa, iptables, mapování adres, NAT, omezování provozu, protokol, python, QoS. SFQ, server, směrovač, směrovací tabulka, přenosová rychlost, TBF, parametr tc, TCP, třída, tvarování provozu, UDP

Abstract

The diploma works deals with the proposal and implementation of wireless networks and servers, which control the data flows into the internal network and back to the external network. Next works is configuration the server and create static routing tables. The wireless network also includes UNIX-based routing elements that are located on the internal wireless network. These devices have also static set routing tables. Internal network devices are a combined form of Ethernet and wireless transmission.

The basic requirement is partition guaranteed bandwidth between clients in the internal network with respect to the distribution of certain services. The guaranteed speed allocation is done when the network cable is connected to the network card server. The server further controls network traffic and classifies it according to priorities into classes. In addition, the network is used for network traffic shaping on all local directional controllers.

Key words

Algorithm, address mapping , bandwidth, , class, CBQ, cycle, ethernet, filter, HTB, IP address, iptables, NAT, traffic constraint, protocol, python, QoS. SFQ, server, router, routing table, TBF, tc parameter, TCP, traffic shaping, UDP, wireless network

Seznam použitých symbolů

Symbol	Jednotky	Význam symbolu
Vp	Mbit/s	Přenosová rychlost
f	Hz	Frekvence
t	s	Čas

Seznam použitých zkratek

Zkratka	Význam
AF	Assured Forwarding
CBQ	Class-Based Queueing
CIDR	Classless Inter-Domain Routing
CoS	Class of Service
DNAT	Destination Address Translation
DNS	Domain Name System
DS	Differentiated Services
DSCP	DiffServ Code Point
ECN	Explicit Congestion Notification
EF	Expedited Forwarding
ELSR	Edge Label Switching Router
FIFO	First-in First-out
FTP	File Transfer Protocol
HTB	Hierarchical Token Bucket
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IANA	Internet Assign Numbers Authority
ICMP	Internet Control Message Protocol
IMQ	Intermediate queueing device
IP	Internet Protocol
IPP	IP precendce
ISP	Internet service provider
LAN	Local Area Network
LDP	Label Distribution Protocol
MPLS	Multiprotocol Level Switching
MTU	Maximum transmission unit
NAT	Network Address Translation

OSPF	Open Shortest Path First
PRIO	Priority Queueing
QoS	Quality of Services
RED	Random Early Drop
SFQ	Stochastic Fairness Queueing
SLA	Service level agreement
SNAT	Source Address Translation
SSH	Secure Shell
TBF	Token Bucket Filter
tc	Traffic Control
TCP	Transmission Control Protocol
TOS	Type of Service
TTL	Time To Live
UDP	User Datagram Protocol
WAN	Wide Area Network

Seznam použitých termínů

Termín	Význam termínu
Bandwidth	Přenosová rychlost
Gnuplot	Výraz pro utilitu pro tvorbu grafů
Mapování IP	Překlad IP adres z vnějších na vnitřní a naopak
Queueing disciplines	Řídící disciplíny
Real-time tok	Provoz v reálném čase
Routing	Směrování
Routing table	Směrovací tabulka
Subnet	Podsíť
Traffic shaping	Tvarování provozu dle zvolených parametrů

Obsah

1 Úvod

Poskytovatelé internetového připojení zejména ti, co se zabývají bezdrátovým potřebují pro svoji síť zajistit vhodné podmínky pro optimální přenos.

Internetoví provideři (ISP) si zakládají zejména na přenosové rychlosti a funkci základních služeb. Pro potřebu přenosu vyšší přenosové rychlosti od vysílače ke klientské stanici se využívají různé způsoby. Jako prvním způsobem jsou bezdrátové zařízení, které vysílají ve vyšších frekvencích, než je standartní pro ISP, jako například frekvence 5 GHz a 2,4Ghz. Dnes se běžně využívají bezdrátové antény vysílající na frekvencích například 10 GHz, 17Ghz a výších. Dalším způsobem je vhodně volit, dle lokality způsob vysílání bezdrátového signálu do prostředí. Pro prostředí, kde je menší počet vysílacích stanic, je možné použít vysílací stanice pro všesměrové vysílaní. Ovšem ve většině případů se dnes již musí, z důvodu většího počtu vysílacích stanic používat stanice pro směrové nebo sektorové vysílaní bezdrátového signálu.

Samotné přivedení přenosové rychlosti ke klientské stanici není vše. V tomto okamžiku je třeba řešit také síť v klientské stanici. Klientskou stanici je myšlena stanice, která využívá částečné možnosti nastavit kvalitu služby. V klientské stanici jsou vedena všechny zařízení uvnitř klientské sítě, které se v některých případech nechovají korektně. Například se může stát, že některé zařízení uvnitř sítě klienta bude omezovat ve využívání přenosové rychlosti ostatní zařízení generováním vysokého množství požadavků. Takové zařízení je vhodné v klientské stanici vhodně nastavit. Nejčastěji se takovým zařízením přidělí určitá přenosová rychlost.

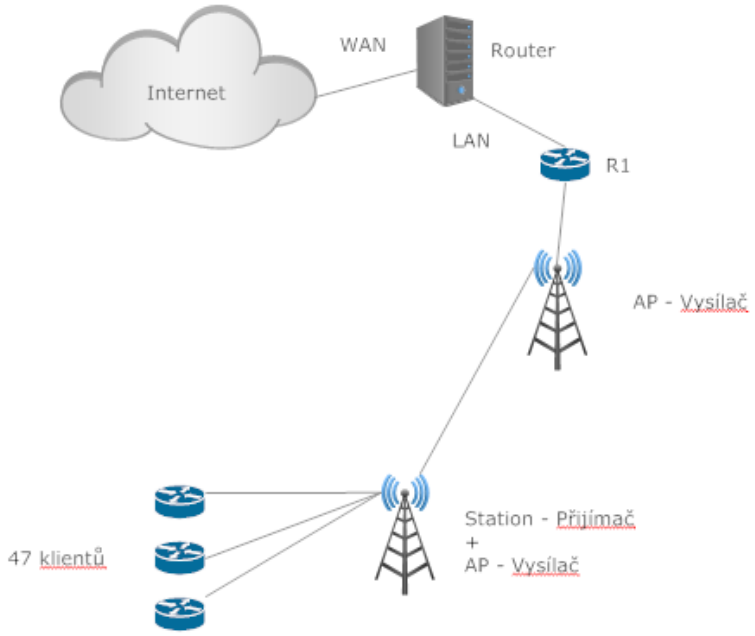
Dále je potřeba v síti zajistit každému klientskému zařízení určitou kvalitu služby. Tím je myšleno, aby všechny klientské stanice měly stejné podmínky pro využívání služeb dle vlastního výběru. Pro splnění této podmínky je zapotřebí vhodně zvolit způsoby rozdělení nejen přenosové rychlosti ale také rozdělení jednotlivých služeb do skupin dle priorit. Nastavení kvality služeb je tedy dalším krokem pro vytvoření spolehlivěji pracující sítě. V kvalitě služby budeme určovat priority jednotlivých služeb. Některé služby je lepší z časového hlediska zpracovávat dříve, než ostatní. Takovéto služby jsou citlivější na rychlost zpracování a měly by se zpracovávat dříve než služby jiné. Pro vytvoření takových pravidel se používá tvarování provozu. Zde je možné nastavit jednotlivým službám dle libosti jejich časové priority na zpracování požadavků a také přenosová rychlost, která jim bude rezervována.

Cílem této práce je vytvořit síťový systém pravidel, který bude obsluhovat zhruba 900 klientských stanic s využitím dostupných nástrojů a disciplín pro omezování a tvarování provozu s přiřazenou šířkou pásma v prostředí UNIX. Konkrétně konfigurací serveru, který bude všechny potřebné kvality služby zajišťovat.

2 Bezdrátové sítě

2.1 Rozdělení bezdrátových sítí

Bezdrátové sítě můžeme rozdělit podle několika typů. Na bezdrátové sítě WLAN (Wireless Local Area Network) jako například Wi-Fi, na sítě WPAN (Wireless Personal Area Network) jako například Bluetooth, na mobilní bezdrátové sítě jako GSM a jiné. Jelikož tato diplomová práce pojednává o bezdrátových sítích WLAN a to především o venkovních Wi-Fi sítích, zaměřme se na tento okruh. Bezdrátové sítě můžeme dělit například na licencované čili placené a nelicencované čili volné používané. Příklad takovéto sítě lze vidět na obrázku 2.1.



Obrázek 2.1: Schéma struktury venkovní WLAN bezdrátové sítě

2.2 Základní komponenty bezdrátové sítě

Mimo rozdělení mezi vnitřními a vnějšími sítěmi, můžeme sítě rozdělit dle jednotlivých komponent, typu sítě a dle používané rádiové frekvence.

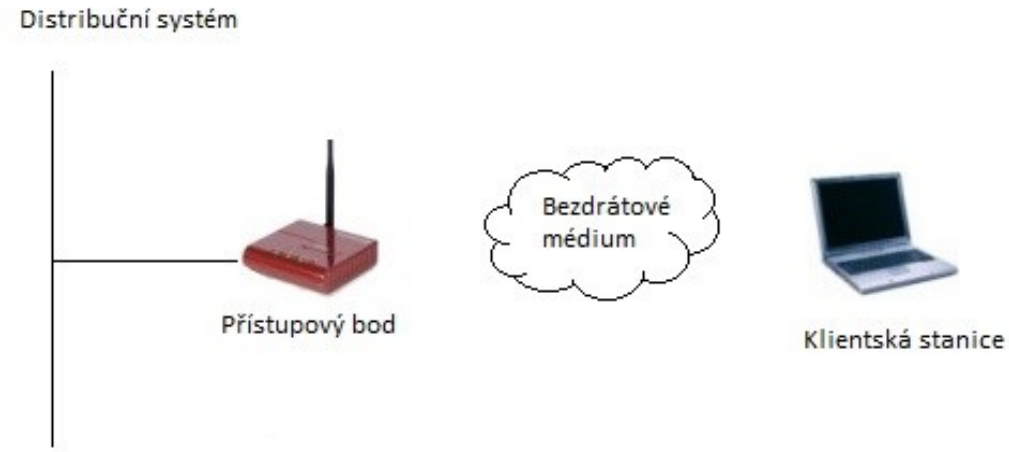
2.2.1 Základní komponenty sítě 802.11

Pro lokální bezdrátové sítě se využívá zkratka WLAN (Wireless Local Area Network). WLAN se svými vlastnostmi podobá Ethernetu, avšak WLAN se musí vyrovnávat s jistými problémy v rádiovém prostředí. Kvůli využití nastavení rozhraní pro bezdrátový přenos přidávají tak mnoho různých funkcí pro nastavení navíc oproti samostatnému Ethernetu.

Myšleno zejména v oblasti managementu sítě. Schéma základních komponent lze vidět na obrázku 2.2.

Tyto sítě obsahují čtyři hlavní druhy fyzických komponent:

Distribuční systém
Přístupový bod (AP access point)
Bezdrátové médium
Station (koncová přijímací strana bezdrátového zařízení)



Obrázek 2.2: Schéma komponent bezdrátové sítě

2.2.1.1 ***Distribuční systém***

Distribuční systém je logická komponenta standardu 802.11. Používá se k přesměrování datového toku na stanici dle aktuální polohy. Distribuční systém je řešen většinou jako kombinace síťového mostu a distribučního média, kde jsou data přenášena mezi přístupovými body. Většinou je páteřní sítí tohoto systému Ethernet.

2.2.1.2 ***Přístupový bod***

Přístupový bod je zařízení označující se většinou jako AP (Access Point). Tento bod si můžeme představit jako zařízení, které vysílá bezdrátový signál. Na tento signál se můžou ostatní zařízení připojit. Příkladem může být například bezdrátový směrovač. Přístupový bod je zařízení, jehož nejdůležitější funkcí je přemostění komunikace mezi kabelovou a bezdrátovou sítí. Dále zajišťuje i provádění autentizace jednotlivých klientů, kteří se chtějí připojit k tomuto bodu. Mimo to zajišťuje i šifrování, bezpečnost komunikace a další činnosti.

2.2.1.3 ***Bezdrátové médium***

Bezdrátové médium je ve WLAN sítích téhož významu jako v sítích Etheret. V sítích Ethernet jsou médium kabelové rozvody, přes které jsou propojeny síťové prvky. Ve WLAN sítích je to vzdušný prostor. Pro sítě 802.11 jsou specifikovány rádiové frekvence od 2,4 GHz po několik GHz.

2.2.1.4 ***Station (klientská stanice)***

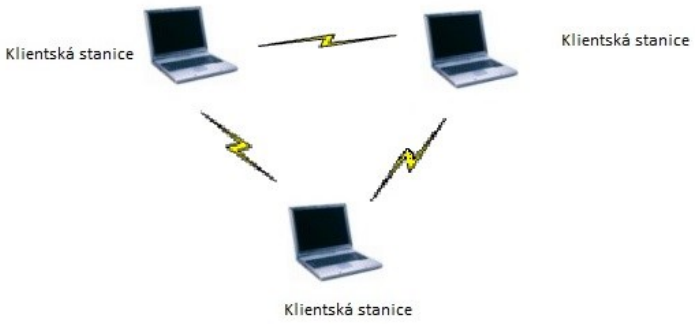
Klientskou stanicí je myšleno jakékoli zařízení, které umí přijímat a vysílat data přes bezdrátové médium. Můžeme si pod tímto pojmem představit zařízení jako například notebook, mobilní telefon, PDA, tiskárny s bezdrátovou síťovou kartou a další. Některé zařízení, které obvykle nejsou vybaveny klientskou stanicí můžou být o tuto stanici rozšířeny. Například externí bezdrátovou kartou a klientskou stanicí se může stát například i stolní počítač.

2.2.2 Typy sítí

Skupiny klientských stanic vytváří v bezdrátových sítích různé typy zapojení sítí. Klientské stanice jsou označovány jako BSS (Basic Service Set). Je to skupina klientských stanic, které spolu komunikují pomocí bezdrátového spojení. Oblast této komunikace se nazývá BSA (Basic Service Area). Podle způsobu komunikace rozdělujeme typy sítí na dva typy.

2.2.2.1 ***Ad-hoc sítě***

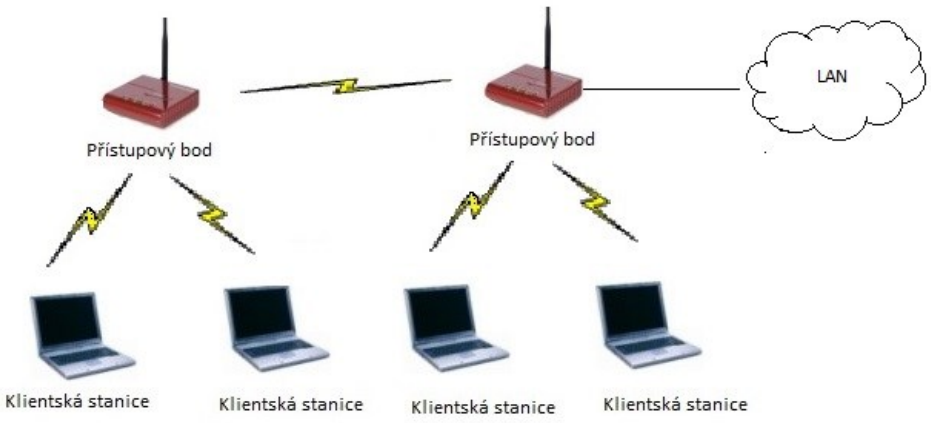
V těchto sítích jsou připojené klientské stanice k sobě bez použité přístupového bodu AP. Jednotlivé klientské stanice spolu komunikují navzájem a musí být v přímém rádiovém dosahu. Bez použití přístupového bodu jsou kladeny velké nároky na samotné klientské stanice. To z toho důvodu, že musí udržovat spojení s ostatními klientskými stanicemi. Tento typ zapojení do sítě se používá nejčastěji do několika metrů. Příkladem takového zapojení může být například propojení několika notebooků v jedné místnosti. Z toho důvodu se tento typ zapojení téměř nepoužívá. Příklad takové sítě lze vidět na obrázku 2.3.



Obrázek 2.3: Příklad zapojení Ad-Hoc sítě

2.2.2.2 ***Infrastrukturní sítě***

V infrastrukturních sítích je způsob zapojení vymezený. Klientské stanice se připojují na jednotlivé přístupové body AP. Tyto přístupové body jsou síťové prvky nejčastěji typu směrovač. Slouží pro propojení mezi bezdrátovou a ethernetovou částí sítě. Tyto směrovače dokáží komunikovat s větším množstvím klientských stanic i mezi jinými síťovými prvky navzájem. Je tedy možné mít rozlehlejší síť se snadnou konfigurací, jelikož o komunikaci se starají směrovače neboli jednotlivé přístupové AP. Tím pádem na klientských stanicích odpadá složitá práce s konfigurací sítě a připojení na ostatní klientské zařízení. Výhodou je celková centrální správa sítě ne jednotlivých přístupových AP. Klientské stanice tedy nemusí být v přímém rádiovém dosahu. Požadavkem je, aby byly v dosahu přístupového bodu AP v jedné infrastrukturní síti. Výhodou takové sítě je dále správa všech klientských stanic a možnosti povolit připojení do sítě jen vyhrazeným klientským stanicím. Také je zde možné síť optimálně zabezpečit a dle možnosti přístupového bodu AP využít i jiné síťové funkce jako například kvalitu služby a další. Tento způsob zapojení do sítě je využíván nejvíce. Především je to díky flexibilitě konfigurace a různým možným změnám v síti, které lze spravovat samostatně na jednotlivých přístupových bodech AP. Tento způsob zapojení se používá i ve venkovních sítích, kde jsou ale použity antény s vyšším ziskem. Příkladem může být vysílací strana jako přístupový bod AP a přijímací strana jako station klientské zařízení. Další možností je využití síťové mostu (bridge). Tento most může tvořit více přístupových bodů, které jsou bezdrátově spojeny pomocí různých typů antén. Ve venkovních podmínkách lze bezdrátově spojit přístupový bod AP a klientskou stanici station až na desítky kilometrů. Příklad zapojení infrastrukturní sítě uvnitř budovy je uveden na obrázku 1.4.



Obrázek 2.4: Hierarchický strom tříd provozu

2.3 QoS v bezdrátových sítích

Jedny z hlavních starších metod pro řízení QoS v bezdrátových sítích popisuje standard IEEE 802.11e. Tento standard byl zaveden v roce 2005. V této práci není využíván. Více o tomto standardu zde [1].

Ve venkovních sítích, o kterých pojednává tato práce, je možné využívat kvalitu služby QoS (Quality of Service) několika způsoby. První způsob je spravovat kvalitu služby dle typu druhu stanice. Podle typu stanice můžeme využívat rozšířené nebo základní nastavení kvality služby. Někteří výrobci ve svých směrovačích nebo přístupových AP mají možnost řídit kvalitu služby buď na úrovni přivedené přenosové rychlosti, řízení přenosové rychlosti na jednotlivé klientské stanice nebo možnost nastavit rozsáhlé složitější politiky blížící se možnostem serveru. Na vysílacím prvku přístupovém AP ve venkovní bezdrátové síti můžeme tedy řídit přenosovou rychlost na jednotlivé klientské stanice. Ve většině případů jsou tyto jednotlivé klientské stanice připojeni pomocí bezdrátové přijímací antény. Klientská přijímací anténa většinou dodává přenosovou rychlost pro menší LAN síť. Další možností jak řídit provoz v bezdrátových sítích je lokálně u každého klientského zařízení a tím mít pod dohledem lokální síť LAN. Toto řízení provozu ve vnitřní síti klientské stanice je velmi využíváno. V případě kolize ve využívání přenosové rychlosti v lokální síti klientské stanice, lze tento problém vyřešit pomocí omezovacích vlastností. V některých případech se může stát, že jedno klientské zařízení spotřebuje přenosovou rychlost přiřazenou celé síti LAN. Můžeme jej v klientské přijímací stanici řídit tak, aby se přenosová rychlost rozprostřela optimálně i ostatním členům lokální sítě LAN. [1][23]

3 Omezování provozu a kvalita služby

Při práci ISP (Internet service provider) je důležité mít kontrolu ve vlastní síti nad přenosovou rychlost a kvalitou služby. Jako prvním krokem je zapotřebí mít dohled nad jednotlivými klientskými stanicemi. Proto zaručení této možnosti se využívá takzvaného omezování provozu. Pod pojmem omezování provozu si můžeme představit například přiražení určité přenosové rychlosti mezi vysílací stanici a klientskou stanicí. Ovšem při větším počtů klientských stanic na jednu vysílací stanici vzniká větší počet různých požadavků na zpracování určitých služeb. Důrazný vliv na tyto služby má rodina protokolu TCP/IP na nichž služby pracují. Tyto protokoly zajišťují určitý typ doručení požadavků od zdroje k cíli. Detailněji popsáno ve zdroji.

Jak bylo zmíněno výše, pří větším počtů klientských stanic je třeba optimalizovat provoz. Pro optimalizování provozu se využívá přiřazení určitých priorit dle typu služeb. Takový provoz se označuje pod anglickým výrazem „traffic shaping“ neboli tvarování provozu.

3.1 Omezování provozu

Určenou přenosovou rychlost můžeme zajistit na mnoha místech v síti. Jedním z míst je hraniční zařízení nazývané server, na jednotlivých vysílacích stanicích nebo na klientských stanicích. Na těchto místech lze využívat kvality služeb (Quality of Service – zkráceně QoS). Více informací lze nalézt zde[3]

Účinně můžeme tedy vykonávat tvarování provozu pouze na hraničních zařízených. Hraničním zařízením je myšleno zařízení nacházející se mezi dvěma sítěmi. Příkladem může být server, vysílací stanice nebo klientská stanice. S pomocí tvarování provozu jsme schopni ovlivnit způsoby zpracovávání jednotlivých požadavků různých služeb.

Tvarování provozu nám zajišťuje:

- Efektivnější využití omezené šířky pásma
- Snížení doby odezvy a zvýšení dostupnosti služeb
- Lepší kontrolu na provozu sít

3.2 Rozdělení provozu

Provoz dělíme dle jeho charakteristických vlastností. Různé typy provozů generují různý počet požadavků, avšak u některých provozů není čas na zpracování tak důležitý a u některých ano. Podle charakteristických požadavků můžeme provoz rozdělovat na různé typy. Tyto typy určují nejen vlastnosti určitého provozu, ale jsou v něm přiděleny služby, které jsou v závislosti na času zpracování závislé.

3.2.1 Typy provozu a jejich rozdělení

Pod objemným provozem si lze představit provoz o velkém objemu dat přesouvaného z jednoho místa na druhé. V takovém případě nezávisí na tom, zda data přijímáme nebo odesíláme ale důležité je, že v tomto případě dochází k zahlcení přenosové rychlosti mezi zdrojem a cílem. Při takovém to využití sítě je bez vhodného managmentu i možná saturace sítě, což může nepříznivě ovlivnit služeb.

Pro tento případ potřebujeme využít nějakým způsobem tvarování provozu, aby nám nedocházelo k saturaci nebo ovlivnění ostatních služeb. Jedním z důvodů proč je tato vlastnost v síti zapotřebí, je například potřeba využívat služby s vyšší prioritou.

Služby s vyšší prioritou na zpracování můžeme označit jako interaktivní. Příkladem takovéto služby je například SSH, telnet, SSL nebo HTTP. Tyto služby využívají pro svou komunikaci krátké zprávy typu žádost/odpověď. Tento případ se týká především služeb, jako je například online nakupování nebo surfování na internetu. Pravdou je, že pro tyto služby není potřeba velká část přenosové rychlosti. Pří využití plné kapacity přenosové rychlosti pro jeden typ provozu může dojít k ovlivnění časového zpracování služeb ostatních. Z tohoto důvodu musíme zajistit optimální rozdělení přenosové rychlosti pro určité služby tak pro objemnější provoz.

Do oblasti služeb náchylných na využití přenosového rychlosti patří služby, které lze zařadit do časově náročných a časově nenáročných. Mezi těmito dvěma druhy provozu je prioritní shoda, že oba typy provozu jsou závislé na jejich časovém doručení. Rozdílem mezi nimi je, že u časově nenáročných služeb můžeme zpomalit dobu pro jejich doručení, aniž by zpomalení mělo zásadní vliv na jejich funkci. Samozřejmě není jedno, kdy bude požadavek u časově nenáročných služeb doručen, ale můžeme si zde dovolit malé zpoždění. Díky tomu máme více času na zpracování provozu časově náročného na zpracování. Mezi služby, které patří do časově náročných jsou například IP telefony, real-time audio nebo video (tzv streamy) nebo DNS požadavky. U těchto požadavků je potřeba zajistit nejvyšší prioritu, protože zde jen malé zpoždění zásadně ovlivní jejich funkci.

3.3 Klasifikace provozu

Před tím než začneme ovlivňovat provoz, musíme ho však klasifikovat a zjistit o který typ provozu se jedná. Proto bychom měli začít s identifikací provozu. Dále pokračovat přes klasifikaci provozu a na základě klasifikace navrhnout vhodné řešení pro řízení provozu.

3.3.1 Definice koncového zařízení

Každé zařízení na hranici s jinou sítí můžeme definovat jako hosta. Toto zařízení je tzv. uzel sítě a musí mít podporu protokolu IP. Takovéto zařízení má přidělenou svojí MAC adresu což je jednoznačný identifikátor zařízení. Dále je mu přidělena IP adresa a maska sítě. Dle těchto identifikačních informací může server (Traffic Shaper) klasifikovat svůj provoz pro jednotlivé služby. Jelikož IP paket obsahuje zdrojovou a cílovou adresu účastníků komunikace, je možné zaměřit se na zdroj, cíl nebo taky na obě adresy současně.

3.3.2 Typy portů

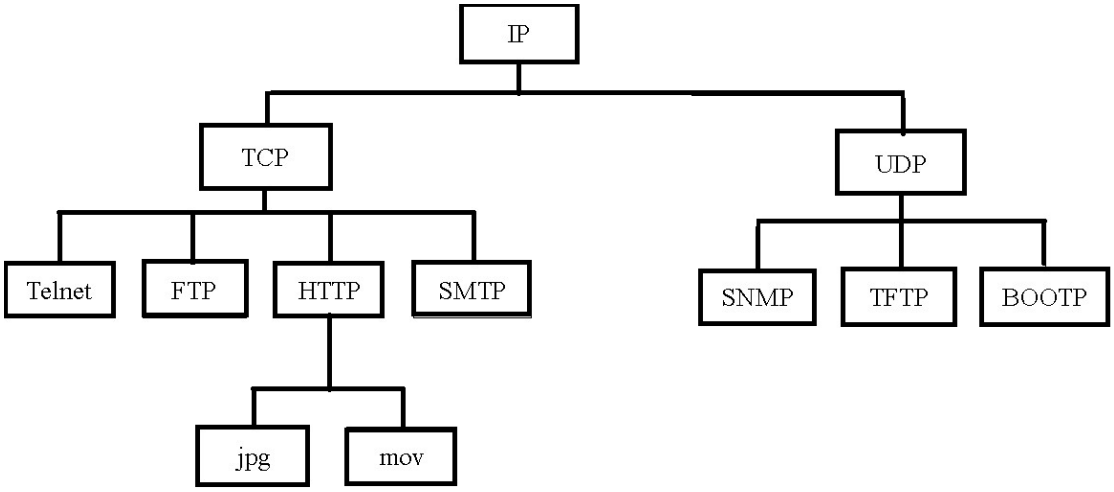
Jednotlivé služby v síti využívají specifických portů ke komunikaci. Každá služba má jedinečné číslo portu. Důvodem je, aby v provozu bylo možné konkrétní služby rozeznat. Tyto čísla portů jsou přiděleny každé službě společností Internet Assign Numbers Authority (IANA). IANA pro ně definuje výraz tzv. „well-known port numbers“ což značí známé čísla portů. Některé z těchto základních služeb jsou udány níže v tabulce č. 3.1. Kompletní přehled všech portů z rozsahu 0 – 65355 je možné vyhledat na Internetu.

Tabulka 3.1: *Standartní služby a jejich porty*

Služba	Port
ECHO	7
FTP(Data)	20
FTP(Data)	21
SSH	22
TELNET	23
SMTP	25
NTP	123
DNS	53
HTTP	80
POP3	110
IMAP	143
SNMP	161
HTTPS	443

3.4 Hierarchický strom tříd provozu

Při definování tříd provozu, uspořádáváme provoz do vztahu rodič -> potomek. Tímto hierarchickým uspořádáním dostaneme přehledný strom tříd provozu, v kterém je jasně vidět vzájemná závislost tříd. Čím níže se potomek v třídě nachází, tím specifičtější kritéria pro něj platí.



Obrázek 3.1: *Hierarchický strom tříd provozu*

Při klasifikaci třídy procházíme celý hierarchický strom tříd od kořene k listům a hledáme v něm stejnou charakteristiku. Při nalezení shody v rodičovské třídě pokračujeme v prohledávání jejích potomků pro hledání bližších informací. Prohledávání hierarchického stromu třídy provozu vyhledává tak dlouho, dokud nenajde provoz, který odpovídá charakteristice toku dat.

Například pokud budeme hledat tok dat pro protokol Telnet na obrázku č. 3.1, tak budeme postupovat následujícím způsobem. Nejprve vstoupíme do rodičovské třídy IP. V něm vyhledáme podtřídu čili potomka TCP jelikož víme, že Telnet je spojově orientovaný. V této třídě už hned vidíme protokol Telnet. Takže v této třídě můžeme už aplikovat příslušné politiky pro traffic shapping.

3.5 Politiky

Při aplikaci politik musíme dbát na jejich správné rozdělení dle požadavkům služeb. V sítích je obvyklým jevem zatížení celé sítě jediným zařízením. V takovém případě může dojít k zahlcení veškerého provozu z důvodu plného využití přenosové rychlosti. V takovém případě se může objevit zpoždění při zpracovávání služeb náchylných na dobu doručení do cíle. Řešením této situace je použití politik pro jednotlivé služby. Využívá se k tomu metoda pro traffic shapping, která nám umožňuje rozdělit přenosovou rychlost. Rezervovaná přenosová rychlost je poté využita k přidělení určité službě. Pomocí rozdělení přenosové rychlosti mezi služby dojde ke spolehlivějšímu fungování služeb v síti. Nenastane totiž okamžik, kdy některé zařízení v síti využije přenosovou rychlost rezervovanou pro naše potřeby. Je tedy výhodné využívat politiky pro časově náročné k zajištění optimálního zpracování požadavku služby a jeho odeslání, v závislosti na čase.

Základní politiky pro traffic shapping jsou založeny na:

- **Přenosová rychlost**
- **Priorita**
- **Zahození**
- **Ignorování**
- **Zamítnutí přístupu**

Samotné politiky jsou založeny na úrovni důležitosti. Tyto politiky fungují na principu rezervace přenosové rychlosti pro jednotlivé typy přenosů. V případě objemných provozů, které zahlcují síť je využití politik nezbytné. Více se lze dočíst v [4].

3.6 Kvalita služby

Kvalita služby zkratkou QoS (Quality of Service) je pojem, kterým v počítačových sítích označujeme zabezpečení potřebných vlastností sítě. Ty nám zaručují doručení dat uživateli v potřebné kvalitě podle daných pravidel. Kromě rozhodování kam daný paket posílat je třeba také rozhodnout v jakém časovém rozpětí jej poslat. Pro dosažení co nejefektivnějšího řešení kvality služby je nezbytné, aby se mechanismy QoS implementovaly po celé trase přenosu od zdroje k cíli. Nejčastějším nasazením QoS je na druhé a třetí vrstvě modelu OSI pomocí protokolu IP. QoS však není jediným nástrojem pro řešení problémů se zajišťováním kvalitativních parametrů přenosu. Používá se celá řada QoS nástrojů. Jejich názvy se můžou lišit v závislosti na tom jakým způsobem se použíjí.

Můžeme je rozdělit do následujících kategorií:

- **Classification and marking** - klasifikace resp. rozlišení a označení provozu
- **Congestion management (Queuing)** - správa zahlcení pomocí front
- **Shaping and Policing** - správa datového toku, omezování a vyhrazování
- **Congestion avoidance** - nástroje pro zabránění zahlcení, např. selektivním zahozením TCP paketu
- **Link-efficiency** - správa efektivního využití linky, obvykle se týká pomalých linek, komprese nebo fragmentace paketů
- **Signaling** (RSVP, QPPB)

3.6.1 Best Effort

První služba v síti, v které se neřeší priorita při přenášení paketů. Dalo by se říct, že tato metoda využívá největší snahy doručit pakety co nejrychleji a nejefektivněji ze zdroje do cíle. Nebude se rozlišovat, zda se jedná o důležité pakety například hlasového nebo streamového hovoru či videa. Takové nastavení však nezaručuje žádnou kvalitu služby. Síť tohoto typu bude nekonečně dlouho zahlcena. V situaci kdy dojde místo v bufferech se začnou pakety(- tail drop). Tato síť není příliš vhodná, jelikož trvale zahlcuje síť nepřetržitým provozem. Jediným vhodným opatřením je použít úroveň správy v operačním systému s pomocí aplikace nebo případně administrativním opatřením.

3.6.2 **Integrated Services**

Nástroje Integrated Serives (dále jen IntServ) definují signalizační proces, pomocí kterého mohou jednotlivé datové provozy požadovat rezervaci přenosovou rychlosta požadovaného zpoždění. Jedna se o signalizaci od zdroje k cíli. Zde se zajišťuje rezervaci přenosové rychlosti a zpoždění po celé trase od zdroje k cíli. Aby bylo možné garantovat přenosovou rychlost a zpoždění, musí architektura zajišťovat minimálně dvě funkce. První je Resource reservation tzv. (rezervace zdrojů) a Admission control tzv. (řízení přístupu). Rezervace zdrojů signalizuje jednotlivým síťovým komponentám, kolik přenosové rychlosti a jaké zpoždění požaduje pro přenos od zdroje k cíli. Za to řízení přístupu rozhoduje, zda takto signalizované požadavky akceptuje nebo zamítne. V současnosti je architektura IntServ v prostředí sítí realizována a implementována protokolem RSVP (Resource Reservation Protocol). Příkladem je následující chování. Vytvoří se požadavek na rezervaci pásma 512 kb/s se zpožděním Low Delay. Tento požadavek obdrží první směrovač po cestě, který rozhodne, zda je možné požadavku vyhovět. Každý ze směrovačů si takto vyhradí rezervovanou část přenosové rychlosti pro jistý datový provoz. Při doručení do cíle se požadavek akceptuje a posílá zpět zprávu o potvrzení. Bohužel RSVP má i některé nevýhody. Je to problematická škálovatelnost, kdy rezervace jsou prováděny pro jednotlivý datový provoz a pro tento jsou také zasílány periodické obnovovací rezervační zprávy. Tyto problémy je možné z části překonat výkonem síťových komponent.

3.6.3 **Differentiated Services**

Diferencované služby Differentiated Services (dále jen DiffServ) jsou mechanismy sloužící k rozdělení služeb, dle jejich nároků na síť. Jedná se o následovníka IntServ architektury. IntServ architektura je z hlediska svého konceptu postavena na signalizaci, zato DiffServ je model postaven na myšlence kategorizace provozu do jednotlivých tříd (CoS – Class of Service), pro kterou jsou nástroji QoS zajištěny kvalitativní parametry přenosu. Tyto parametry jsou definovány v rámci DiffServ. Jsou však uplatňovány na jednotlivých směrovačích nezávisle. DiffServ architektura slouží výhradně pro upřednostňování určitého provozu před jiným. Celý proces je možné popsat následovně. Paket přichází na hraniční směrovač, kde je označen odpovídající značnou v IP hlavičce. Ukládá se do tzv. položky v TOS (Type of Service), které je označováno jako pole DS (Differentiated Services). Jeho strukturu Znázorňuje tabulka č. 3.2. Z tabulky vyplývá, že prvních 6 bitů čili bity 0 až 5 se označují jako DSCP (DiffServ Code Point). DSCP slouží pro označení třídy provozu. Zbylé dva bity ECN (Explicit Congestion Notification) jsou volitelně využívány pro informaci přetížení sítě, mezi dvěma uzly bez ztráty paketů.

Tabulka *3.2*: *DS (Typ služby) V IP hlavičce*

Bity	0-5	6.7
0	DSCP	ECN

Pole DSCP nabývá hodnoty 0 až 63 a to podle typu aplikace služby Uvnitř DSCP je také tzv. IP precendce (IPP) o velikosti 3 bity. Dříve se používala pouze hodnota IPP, později se pole pro metodu DiffServ rozšířilo na DSCP, které je zpětně kompatibilní s IPP. V tabulce 3.3 jsou tyto informace zobrazeny. Více informací o DSCP naleznete v [4].

Z hlediska dnešní doby je DiffServ nejvíce využívána, protože poskytuje specifické nastavení kvality služeb, jaké je zapotřebí. Velmi často se uplatňuje strategie rozdělení přenosové rychlosti do několika málo tříd. Určitým třídám vyhradíme vlastní přenosovou rychlost pro každou službu zvlášť.

Tabulka *3*.3: *Hodnoty IPP a DSCP a k nim přiřazené typy aplikace*

IPP	DSCP	typická aplikace
7		rezervováno
6	48	routing
5	46	hlas
5	34	video konference
4	32	streamované video
3	26	mission critical data
3	24	call signaling
2	18	transaction data
2	16	network managment
1	10	bulk data
1	8	scavenger
0	0	Best-effort data

Model DiffServ rozlišuje dále vazby mezi označeným paketem a jeho zpracováním v každém síťovém zařízení. Každé síťové zařízení má na třetí vrstvě jasně definován způsob, jak zachází s jednotlivými pakety. Jinými slovy chováni v rámci skoku PHB (Per Hop Behavior). Zvolení správného PHB pro daný paket může být určen z různých požadavků na daný síťový uzel, například:

- nároky na vyrovnávací paměti, šířku pásma
- požadované parametry zpoždění, kolísaní zpoždění, ztrátovosti

V mezích DiffServ byly definovány tři kategorie jak zacházet s pakety. Příkladem jsou tři typy PHB:

- **urychlené předávání (Expedited Forwarding**) EF – Urychlené předávání zajišťuje minimální ztráty paketů, maximální zpoždění a jeho kolísání. Garantována je v tomto případě šířka pásma a je vhodná například pro hlasové služby.
- **zajištěné předávání (Assured Forwarding**) AF– zaměřuje se na minimální ztrátu paketů, a na služby vyžadující určitou spolehlivost. V tomto případě není brán ohled na maximální zpoždění a díky tomu je možné ovlivnit velikost kolísání. Nehodí se pro služby pracující v reálném čase, kdy je vyžadováno co nejnižší zpoždění.
- služba **best-effort**

Urychlené předávání je náročnější a omezené na určitý počet datových provozů. Zajištěné předávání nám ale nezaručuje přesné dodržení všech parametrů QoS. Tabulka 3.3 shrnuje hodnoty DSCP pro všechny typy PHB

3.6.4 Parametry QoS

Parametry QoS lze reprezentovat jako charakteristiky datových provozů a síťových linek. Zároveň vyjadřují jejich výkonnost, kvalitu, chybovost, ztrátovost paketů, propustnost, latenci, detekci a opravu chyb.

Mezi základní parametry patří:

- **Bandwidth** (šířka pásma) – udává množství bitů přenesených za jednotku času
- **Delay** (zpoždění) – udává čas potřebný k odeslání paketů a jeho přijetí v cíli. Z hlediska zpoždění jsou vhodné malé velikosti fragmentů. To ale způsobuje zatížení sítě, protože se změní rozdíl velikosti hlavičky a daty, které paket obsahuje
- **Jitter** (kolísání velikosti zpoždění) – je definován jako rozdíl mezi jednosměrným zpožděním dvou paketů
- **Packet loss** (ztrátovost paketů) – Datagramy, které jsou ztraceny a nemůžou být obnoveny, vytvářejí v konverzaci mezery. Pokud je ztráta datagramu rozložena náhodně nevede to až tak k významnému zhoršení hlasové kvality

3.7 Kontrola provozu

Kontrola provozu také označovaná z angličtiny jako „traffic control“ probíhá na síťových uzlech v síti. Tato kontrola provozu zásadně ovlivňuje kvalitu služby, protože je jejím základním nástrojem. Jedná se o celou cestu mezi vstupním a výstupním síťovým zařízením. Dějí se zde základní rozhodovací procesy při průchodu paketu. Při doručení paketu, je paket klasifikován na vstupním zařízení. Zde se rozhodne, zda bude paket zahozen, odmítnut, ignorován nebo přijat. Poté co se vykoná činnost klasifikace. Poté je paket buď určen pro toto zařízení a je poslán k vyšším vrstvám nebo může být taky určen pro jiné zařízení. V takovém

případě je paket poslán na jiný síťový uzel a podle toho se vybere vhodné výstupní zařízení. Pokud tomu velí nastavena politika, tak je paket buď zahozen anebo odeslán.

Najdeme zde tři základní označení:

- **INPUT** – označení pro vstupní řetězec
- **OUTPUT** – označení pro výstupní řetězec
- **FORWARD** – označení pro předávací řetězec

Principem je vyhodnocení paketu na vstupním zařízení. Zde je paket zařazen do vstupního řetězce (INPUT). Podle vstupní politiky je klasifikován a zařazen do vstupních front. Po opuštění vstupních front se zjistí, zda je paket určen lokálnímu stroji nebo jinému. Pokud je určen pro lokální stroj, putuje na vyšší vrstvy. Pokud je určen pro jiný stroj, provede se předání tak, že paket putuje do předávacího řetězce (FORWARD). Tento řetězec předá paket dál. Předtím je paket filtrován nebo označkován a je na něj navázáno vlastní směřování (routování). Na výstupních frontách je poté prováděn „postrouting“. Postrouting je spjat s tzv. NAT (Network Address Translation). Dochází zde ke změně IP adresy. V případě využití Maškarády, což je vlastnost NAT, je možné se setkat ještě s pojmy SNAT (Source Address Translation) a DNAT (Destination Address Translation). Při rozhodování co se s daným paketem bude dít dále, je paket poslán do výstupního řetězce a odeslán. U výstupních front můžeme ovlivnit způsob posílání jednotlivých paketů. Může zde například probíhat vlastní kontrola služby. Pakety se na výstupních frontách pochopitelně řadí podle stanovených politik, které můžeme ovlivnit. Tím dosáhneme kontroly nad využívanou přenosovou rychlostí a určení důležitosti paketu. Více se lze dočíst v [20].

3.8 Způsoby využití traffic shapingu

V současné době se na zařízeních využívají distribuce Linuxu. Ty mají jádro, které již plně podporuje různé mechanismy pro tvarování provozu či mechanismy pro řazení paketu do front. Některé známé příklady traffic shappingu jsou uvedeny níže.

- **CBQ** – Class-Based Queueing
- **HTB** – Hierarchical Token Bucket
- **IMQ** – Intermediate queueing device
- **PRIO** – Priority Queueing
- **RED** – Random Early Drop
- **SFQ** – Stochastic Fairness Queueing
- **TBF** – Token Bucket Filter

Existuje tedy spoustu možností tvarování provozu, které můžeme pro naše potřeby využít. Popíšeme si některé z nich. Jejich primární funkce a příklad konfigurace. Všechny výše zmíněné mechanismy jsou využívány delší dobu, takže existuje mnoho různých způsobů kde a jak je využívat. Dnes se nejvíce využívá mechanismus HTB a CBQ. Důvodem

je hlavně to, že HTB dokáže přehledně rozdělit provoz. O tomto mechanismu bude detailně věnována samostatná kapitola níže. Také velmi populární metodou je kombinace více než jednoho mechanismu pro tvarování provozu. Své zastánce má i CBQ proto se dá z dostupných zdrojů najít větší množství informací, kde se zastánci mezi sebou dohadují, který z mechanismů je vhodnější pro praktické využití. Více se lze dočíst v [16].

3.9 IPTABLES

Prvním způsobem jak manipulovat s pakety je přes nástroj IPTABLES. Tento nástroj umožňuje zacházet s pakety podle předem definovaných pravidel. K práci využívá řetězce, které jsme si popsali v předchozí kapitole. V těchto řetězcích je možné definovat pravidla, které provádí vlastní filtrování paketů. Tyto pravidla vytvářejí určité podmínky, kterými musí paket projít. Jelikož je spousta možností jak zacházet s pakety pomocí IPTABLES, uděláme názornou ukázku jednoduššího příkazu, který nám nastíní, jak tento nástroj funguje. Více o iptables naleznete v[15].

Příklad :

```
Iptables –A INPUT –s 10.0.0.10 –j ACCEPT
Iptables –A INPUT –tcp –dport 22 –j DROP
```

Tyto dva příkazy přidají dvě pravidla na konec řetězu INPUT. První pravidlo zachytí pakety se zdrojovou adresou v IP hlavičce (-s 10.0.0.10) a propustí je (-j ACCEPT). Daný paket nebude procházet žádnými dalšími pravidly. Pokud paket první pravidlo nesplní, tak přichází na řadu druhé pravidlo, které nám říká, že jakýkoli TCP paket s cílovým portem 22 (-p –dport 22) vyhoví pravidlu, bude zahozen (-j DROP).

Takto zhruba může vypadat příklad pro pravidla vznikajících v IPTABLES. Ovšem pravidel v tomto nástroji je mnoho a mnoho různých způsobů jak je využít. Také existují i jiné metody, které s pakety zacházejí určitými způsoby. Může se jednat například o zakazování, odmítnutí paketů nebo o jiné typy činností pro práci s pakety. Jelikož je toto téma velice rozsáhlé, tak si zde rozebereme základní pravidla a více o IPTABLES nalezneme v [15].

3.9.1 Základní pravidla IPTABLES

Mezi základní definované pravidla patří :

- **INPUT** – zde je paket zpracován předtím, než se pošle do řetězce ke zpracování směrováním
- **PREROUTING** – slouží pro změnu TOS polí nebo pro přípravu DNAT
- **OUTPUT** – pro filtrování odchozího provozu
- **POSTROUTING** – využívá se při SNAT a ve firewallu než je paket odeslán

3.9.2 Nástroj TC

Dříve jsme se zmínili o tom, jak funguje Traffic Control. Nástroj tc je zkrátka Traffic Controlu a využívá se pro řízení provozu. Pomocí tohoto nástroje lze na uživatelské úrovni vytvářet a asociovat fronty s výstupními řetězci jako například u řetězce OUTPUT. Využívá následujících komponent:

- **Qdisk** – z anglického výrazu „ queueing disciplines“. Tento příkaz qdisk pracuje s přeposíláním dat
- **Třídy provozu** - Tyto třídy rozdělují provoz, s kterým chceme pracovat. Existuje také možnost, že se jedna třída může dělit na více podtříd. Pokud se nedělí na více tříd, je ji přiřazena jedna qdisc, která se stará o zpracování provozu a přenos dat
- **Filtry** – označují se také slovem klasifikátory. Jsou spjaty s třídami a rozdělují provoz do podtříd dané třídy
- **Kontrolor** – Provádí kontrolu, zda filtry třídí provoz správným způsobem

Každému rozhraní je dále ještě přidělena tzv. root qdisc, přes který musí veškerý provoz projít. Počáteční nastavení na rozhraní přikazuje využít známou frontu FIFO. Pokud je potřeba frontu FIFO změnit je možné ji nahradit jinou disciplínou.

3.10 Frontové disciplíny

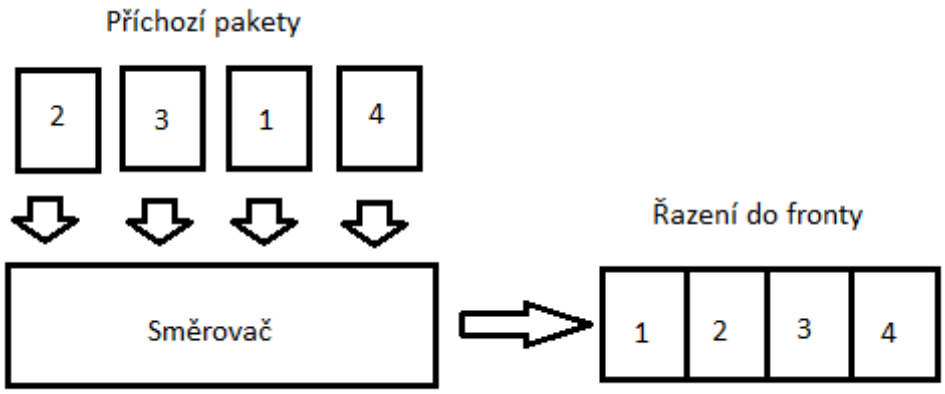
Při příchodu paketu na síťové rozhraní běžně dochází k řazení paketů do front. V těchto frontách se poté pakety filtrují. Zvolí se zde, které pakety budou upřednostněny pro odeslání dále ze směrovače. K této činnosti se používají například role nebo pokročilejší algoritmy. Frontové disciplíny (queueing disciplines) dále dělíme na beztřídní (classless) a třídní (classfull).

3.11 Beztřídní frontové disciplíny

Tyto disciplíny se zakládají na vhodném nastavení přenosového pásma bez aplikace složitých pravidel. Je to nejběžnější využívaný způsob. Problémem je ovšem, že pakety ve frontách nejsou nijak řazeny podle důležitosti. Také zde není možné na výstupní fronty přidat další disciplíny. V tomto případě se používá výhradně pouze pro funkce zařazení, zahození nebo zdržení paket. Příkladem takovéto fronty je fronta FIFO popsaná níže.

3.11.1 **FIFO(First-in First-out)**

Fronta FIFO (First-in First-out) je jedna ze základních disciplín pro řazení paketů do front. Nevyužívá žádných speciálních pravidel nebo algoritmů pro řízení při odeslání paketu z fronty. Jednoduše platí pravidlo, že první příchozí paket do fronty se odešle jako první z fronty do sítě. Toto řešení je vhodné například pro služby typu best-effort. Problém je však, že není možné v této metodě vhodně rozdělit pakety podle jejich druhu nebo priority. Kvůli tomuto nedostatku v této frontě není možné zajistit rozdělení jednotlivých paketů složitějšími metodami. Příklad FIFO fronty je uveden na obrázku č. 3.2.



Obrázek 3.2: Příklad řazení paketů do fronty FIFO

I když nemůžeme aplikovat pokročilejší pravidla pro příchozí pakety, je způsob pakety alespoň částečně řadit. K tomu slouží upravená verze fronty nazvaná pfifo-fast. V této variantě je každý z přijatých paketu na síťovém zařízení řazen do jednoho ze tří kanálů. Zařazení do určitého kanálu je prováděno podle informací z datového pole ToS (Type of Service), které je obsažené v IP datagramu. Jednotlivé kanály jsou typu FIFO. Číslovány jsou v mezích 0,1 a 2. Podle důležitosti paketu jsou podle priorit řazeny do kanálu. Pakety s nejvyšší prioritou budou zařazeny do kanálu 0 a nižší priority potom do kanálu 1 a 2. Důležitost hodnoty v poli ToS ukazuje tabulka č. 3.5.

Tabulka č. 2.5: *Přehled informací v ToS*

TOS	Bity	Význam	Linux	Provoz	Kanál
0x0	0	Normální služba	0	Best Effort	1
0x2	1	Minimalizuj finanční náklady (mfn)	1	Filler	2

0x4	2	Maximalizuj spolehlivost (ms)	0	Best Effort	1
0x6	3	mfn+ms	0	Best Effort	1
0x8	4	Maximalizuj propustnost (mp)	2	Bulk	2
0xa	5	mfn+mp	2	Bulk	2
0xc	6	ms+mp	2	Bulk	2
0xe	7	mfn+ms+mp	2	Bulk	2
0x10	8	Minimalizuj zpoždění (mz)	6	Interactive	0
0x12	9	mfn+mz	6	Interactive	0
0x14	10	ms+mz	6	Interactive	0
0x16	11	mfn+ms+mz	6	Interactive	0
0x18	12	mp+mz	6	Int. Bulk	1
0x1a	13	mfn+mp+mz	4	Int. Bulk	1
0x1c	14	ms+mp+mz	4	Int. Bulk	1
0x1e	15	mfn+ms+mp+mz	4	Int. Bulk	1

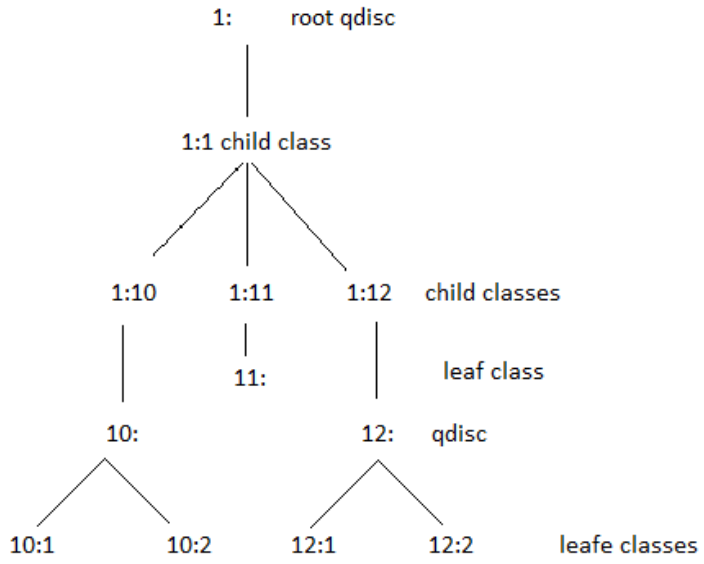
3.11.2 **TBF (Token Bucket Filter)**

Dalším algoritmem využívající pro svou práci princip fronty pfifo je například TBF(Token Bucket Filter). Je to jednoduchá a výpočetně nenáročná disciplína. I když není tento algoritmus schopný využít priorit, dokáže fungovat na principu omezováním provozu. Principem omezování šířky pásma a to s použitím žetonu (token), kterým je konstantní rychlostí plněn zásobník (bucked). Pokud zásobník obsahuje žetony, které odpovídají datovému toku, budou pakety odeslány. V případě neshody budou pakety zahozeny nebo pozdrženy. Více se lze dočíst v [17].

3.12 **Třídní frontové disciplíny**

Třídní frontové disciplíny (Classful queuing disciplines) rozdělují provoz do jednotlivých tříd, které mají své specifika. V těchto třídách jsou pakety rozdělovány podle základních pravidel definovaných správcem. Celý tento systém se nazývá klasifikace provozu. U klasifikace provozu je možné využívat informace obsažené v IP hlavičce paketu, jako jsou například IP adresa, port a další. Vytvořené přesně definované třídy však můžou obsahovat další

vlastní metody qdisc a to jak classfull, tak classless. Tímto způsobem můžou nakonec vzniknout i velmi složité a objemné stromové struktury hierarchie tříd. Každé rozhraní má podle této logiky jednu výstupní qdisc, která je kořenová (root qdisc) a je nastavena na pfifo qdisc. Ke každému qdisc je přiřazený určitý handle. Handle obsahuje dvě části qdisc. První je majoritní číslo a druhé minoritní číslo. Standartním popisem kořenové třídy je hodnota 1:0. To nám ukazuje, že ve většině případů je minoritní qdisc 0. U použití tříd je nutné, aby se shodovalo majoritní číslo s majoritním číslem jejich rodičovských tříd. Ukázka takovéto hierarchie je na obrázku č. 3.3.



Obrázek 3.3: Příklad typické hierarchie tříd v třídních frontových disciplínách

Z výše uvedeného obrázku 3.3 je třeba zmínit, že jádro komunikuje pouze s root qdsic. Dává kořenovému qdisc příkazy na řazení paketů do jednotlivých tříd stromu. Případně také může předat informaci o přeřazení paketu na jiné rozhraní. Daný paket bude procházet hierarchií stromu následovně.

Berme v potaz, že máme paket, který se musí dostat do třídy 10:1. Jeho cesta od kořenového root qdisc bude dle označení tříd následující:

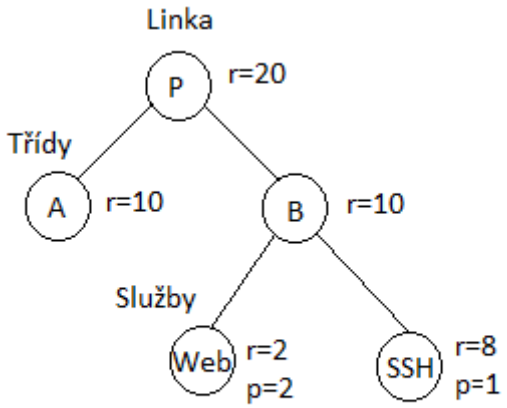
1: → 1:1 → 1:10 → 10: → 10:1

Jeho cíl je tedy 10:1 a paket musí k cíli projít celou hierarchií stromu potřebnou k dosažení cíle. Musí tedy projít všechny potřebné třídy. Případně o zařazení může rozhodnout kořenová třída. V této třídě kde bude použit filtr, který paket zařadí přímo do určitě fronty. Příklad je uveden níže.

1: → 1:10

3.12.1 **Class-Based Queuing (CBQ)**

CBQ je classful omezovací algoritmus. Je pravděpodobně nejsložitějším a nekomplikovanějším algoritmem. Důležité negativa zakončuje svojí složitou konfigurací. Tyto negativní vlastnosti má CBQ hlavně kvůli nepřesným výpočtům. Podporuje stejně jako ostatní disciplíny třídy classful rozdělování provozu do tříd a možnost libovolného vnořování tříd. Dokonce si i jednotlivé třídy můžou půjčovat mezi sebou nevyužitou šířku pásma. Zároveň také dokáže omezit šířku pásma na jednotlivých třídách. Při omezování šířek pásem bohužel nedochází k přesným výsledkům. Představme si, že máme linku s přenosovou rychlostí 10 Mb/s. Tuto linku omezíme pomocí CBQ na 1 Mb/s. Logicky pak znamená, že zbylých 9 Mb/s linky by mělo být nevyužito po dobu 90% času. Pokud není nevyužita, musíme zajistit frontováním paketů její prázdnost po dobu 90% času. Překážkou měření procent času, po které je linka prázdná. Měření času je vcelku složité. Proto CBQ raději odvozuje hodnotu času od počtu mikrosekund, které uběhnou mezi jednotlivými požadavky z hardwarové vrstvy na předání ostatních paketů. Z toho důvodu není disciplína příliš vhodná pro moderní použití, jelikož poskytuje dost nepřesné výsledky. Více informací zde[2].



Obrázek 3.4: Hierarchické sdílení

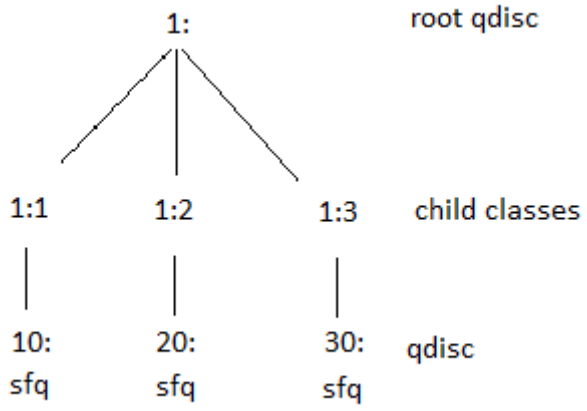
Problémy které se běžně řeší, jsou znázorněny na obrázku č. 3.4. Začněme od shora. Poskytovatel P můžeme si jej představit, jako providera Internetového připojení potřebuje rozdělit linku o velikosti 20 Mb/s mezi jednotlivé linky A a B. Každé lince jak A tak B přiřadí rychlost 10 Mb/s. Každá tato linka má dále specifický požadavek na garantovanou přenosovou rychlost a maximální dobu odezvy. Velikost odezvy záleží na prioritě dané třídy. V našem případě vidíme, že na lince B bude upřednostněna služba SSH před Webem.

Parametry a použití tc:

- **avpkt** – velikost průměrného paketu. Potřebné ke stanovení času, který bude potřeba pro odeslání jednoho paketu. Určený čas je stanoven podílem AVPKT a BANDWIDTH
- **bandwidth** – maximální přenosová rychlost pro zařízení, které je připojeno na frontu. Například ethernetová síťová karta 1 Gb/s, pak bude hodnota 1 Gb/s
- **bounded** – označuje, že třída si nemůže vypůjčit nevyužitou kapacitu třídy svých rodičů. Pokud tato hodnota není specifikována, defaultně se nastaví na hodnoty, kdy si nevyužitou kapacitu vypůjčit může
- **cell** – Velikost kroku, po kterém se bude doba potřebná k odeslání paketu zvětšovat
- **maxburst** – maximální počet bitů, které lze poslat při krátkodobém nárůstu požadavků
- **minburst** – nejmenší možný počet bitů, které je nutno poslat
- **mpu** – minimální počet bitů, které budou poslány v jednom paketu. Pokud je požadavek na odeslání paketu o velikosti menšího počtu bitu, bude paket dorovnán na velikost MPU
- **rate** – Požadovaná rychlost, která má být přidělena dané třídě
- **prio** – nastavuje prioritu dané třídy

Ukázková konfigurace

Mějme linku s hodnotou 40 Mb/s, na které vytvoříme třídy pro provoz SSH, FTP a web. Službě SSH přidělíme 15 Mb/s, FTP přidělíme 10 Mb/s a webu přidělíme 15 Mb/s. Dohromady jako lince bude přiděleno celkem 45 Mb/s. Třídy si můžou navzájem od sebe půjčovat přenosovou rychlost do výše celkové přenosové rychlosti 30 Mb/s, která je přidělena lince.



Obrázek 3.5: Hierarchie CBQ

Příklad vytvoření stromu v terminálu:

```
# tc qdisc add dev eth0 root handle 1: cbq bandwidth 40Mbit
avpkt 1000 ceil 10

# tc class add dev eth0 parent 1:1 classid 1:1 cbq bandwidth
40Mbit rate 15Mbit weight 1Mbit prio 4 ceil 10 maxburst 30 avpkt
1000

# tc class add dev eth0 parent 1:2 classid 1:2 cbq bandwidth
40Mbit rate 10Mbit weight 1Mbit prio 5 ceil 10 maxburst 30 avpkt
1000

# tc class add dev eth0 parent 1:3 classid 1:3 cbq bandwidth
40Mbit rate 15Mbit weight 0.3Mbit prio 6 cell 8 maxburst 30
avpkt 1000
```

Na začátku si vytvoříme kořenový qdisc označený jako 1:0, který bude používat CBQ. Bandwidth v této disciplíně bude nastaven na 100 Mb/s. Velikost průměrného paketu nastavíme na 1000 bitů a hodnotu ceil na 10. K tomuto qdisc jsou připojeny třídy 1:1, 1:2 a 1:3, které reprezentují SSH, FTP a Web. Všechny 3 třídy mají stejnou rodičovskou třídu a to 1:0. První třída bude 1:1 a bude použita pro SSH službu. Tato služba bude mít maximální bitovou rychlost 15 Mb/s s nejvyšší prioritou. Druhá třída s označením 1:2 bude sloužit pro FTP službu. Bitová rychlost bude 10 Mb/s nižší prioritou oproti SSH. A poslední třetí třída bude mít maximální bitovou rychlost 15 Mb/s se stejnou prioritou jako FTP.

Samotná výše zmíněná konfigurace není dostačující. Navíc je třeba přiřadit následující výše vytvořeným třídám disciplíny front. Tyto fronty budou označeny pro každého rodiče a disciplínu následovně:

Rodič -> disciplína fronty

1:1 -> 10:

1:2 -> 20:

1:3 -> 30:

V termínálu pak :

```
# tc qdisc add dev eth0 parent 1:1 handle 10: sfq

# tc qdisc add dev eth0 parent 1:2 handle 20: sfq

# tc qdisc add dev eth0 parent 1:3 handle 30: sfq
```

Nakonec je třeba nastavit používání filtrů pro jednotlivé typy provozu. Dle konfigurace lze usoudit, že filtry budou přiřazeny přímo v kořenové třídě. V tomto případě je použit filtr u32. V prvním příkaze je provoz se zdrojovým portem 22 zařazen pro SSH do třídy 1:1. Druhý

příkaz se zdrojovým portem 21 pro FTP do třídy 1:2. Poslední provoz se zdrojovým portem 80 pro web (http) do třídy 1:3

```
# tc filter add dev eth1 parent 1:0 protocol ip prio 1 u32 match
ip sport 22 0xffff flowid 1:1

# tc filter add dev eth1 parent 1:0 protocol ip prio 1 u32 match
ip sport 21 0xffff flowid 1:2

# tc filter add dev eth1 parent 1:0 protocol ip prio 1 u32 match
ip sport 80 0xffff flowid 1:3
```

3.12.2 Hierarchical Token Bucket (HTB)

Poslední disciplínou je HTB(Hierarchical Token Bucket). Tato disciplína je podobná disciplíně CBQ. Autorem této disciplíny je český tvůrce Martin Devera. Disciplína HTB využívá hierarchická strom tříd provozu a jednotlivým třídám umožňuje nastavit šířku pásma i různou prioritu. HTB je ideálním řešením pro místa, kde je více uživatelů využívajících různých služeb. Těm pak můžeme zaručit potřebnou kvalitu služby s maximální přenosovou rychlostí a využitím neúplného vytížení linky.

Parametry a použití tc

- **Rate** – garantovaný tok pro danou třídu
- **Ceil** – je to označení pro stropovou hodnotu přenosové rychlosti, vyšší rychlosti než ceil nelze dosáhnout
- **Prio** – reprezentuje prioritu třídy
- **Mtu** – maximální velikost paketu
- **Burst** – maximální velikost paměti, kam budou pakety ukládány
- **Cburst** – maximální velikost paměti pro strop

HTB rozlišuje navíc stavy uzlů hierarchického stromu tříd a to z aktuální velikosti datového toku. Tento datový tok se měří v krátkém časovém okamžiku. Označení a jednotlivé mody jsou popsány níže.

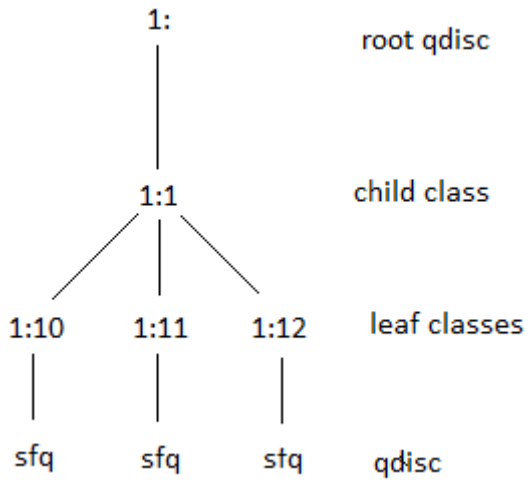
- r - garantovaný tok
- a – aktuální velikost toku
- c – maximální rychlost

Třída zelená - platí $a \leq r$.

Třída oranžová - platí $c > a > r$, v tomto případě třída přesáhla svůj garantovaný tok „r“, avšak si může za určitých podmínek půjčit od rodičů nebo sourozenců. Třída červená - platí $a \geq c$, třída překročila svůj strop a proto není možné odesílat další pakety

Popíšeme si stručně algoritmus, podle kterého si definujeme výběr dalších paketů k odeslání. V tomto algoritmu u všech plných listů v hierarchii stromu si zvolíme takový list, který má možnost si při odesílání paketů vypůjčit volnou přenosovou rychlost od rodiče na nejnižší úrovni. V případě, že je takových listů více, vybere se list s nejnižší prioritou. Pokud bude nadále více listů k využití, vystřídáme je pravidelně mezi sebou podle poměru jejich přenosové rychlosti. Tím zajistíme, že třídy se stejnou prioritou si budou půjčovat pouze od společného rodiče podle jejich přenosové rychlosti. Také bude platit, že třída s vyšší prioritou získá přenosovou rychlost od společného rodiče přednostně.

Půjčování probíhá následovně. Máme strom tříd, kde každá třída má svoji rychlost (rate) a prioritu. Pakety jsou v této třídě rozděleny do toků a přiřazeny do jednotlivých listů stromu, pokud jsou prázdné. Jako první vybereme listy, jejichž kapacita přenosové rychlosti nebyla naplněna. Z těchto listů pošleme pakety s vysokou prioritou a postupně pokračujeme do listu s nižší prioritou. Pokud se překoná šířka pásma na všech listech, celý cyklus se zopakuje. Při tom ale testujeme každý list, zda si může půjčit od rodičů volnou přenosovou rychlost místo testování vlastní přenosové rychlosti. Pokud taková třída není konečný cyklus, pokračujeme k prarodičům třídy a cyklus opakujeme.



Obrázek 3.6 Hierarchie ukázkové konfigurace HTB

Ukázková konfigurace

Pro příklad konfigurace použijeme výše uvedený strom na obrázku č. 3.6. Základem stromu je opět root qdisc 1:0. Dále jsme si vytvořili třídu 1:1v které si nadefinujeme naše třídy. Třídy jsou označeny dle provozu. Třída 1:10 je vyhrazena pro SSH služby, třída 1:11 je vyhrazena pro FTP služby a třída 1:12 je vyhrazena pro web (http). Nyní si uvedeme příklad konfigurace.

```
# tc qdisc add dev eth1 root handle 1: htb default 30

# tc class add dev eth1 parent 1: classid 1:1 htb rate 10Mbit
burst 30kbit

# tc class add dev eth1 parent 1:1 classid 1:10 htb rate 5Mbit
ceil 10Mburst 30kbit

# tc class add dev eth1 parent 1:1 classid 1:11 htb rate 5Mbit
ceil 10Mbit burst 30kbit

# tc class add dev eth1 parent 1:1 classid 1:12 htb rate 5Mbit
ceil 10Mbit burst 30kbit
```

Tato konfigurace nám vytváří třídy postupně od vrchní části stromu z obrázku č. 3.6. V prvním příkazu vytváříme root qdisc. V druhém příkazu vytváříme rodičovskou třídu 1:1 s využitím HTB s přenosovou rychlostí rate 10 Mb/s a maximální velikosti paměti 30kbit. Dále v této třídě máme leaf třídy s rodičem 1:1. Tyto listy jsou definované 1:10, 1:11 a 1:12. Všem listům je nastavená přenosová rychlost rate 5 Mb/s se stropovou rychlostí 10 Mb/s a maximální pamětí 30 kb/s. Všechny 3 služby mají přenosovou rychlost 5 Mb/s. Pokud je některá služba více zaneprázdněna, může si v závislosti na prioritě vypůjčit přenosovou rychlost z daného listu. Pouze ale o velikosti stropové přenosové rychlosti, který určuje parametr ceil.

Z výše uvedeného popisu můžeme vidět, že HTB opravuje, nebo-li zlepšuje nedokonalosti u CBQ. U HTB nedochází k chybným výpočtům a především dokáže využívat nevyužitou přenosovou rychlost svých sourozeneckých listů nebo tříd mezi sebou v závislosti na prioritě. Více informací najdete ve [22]

4 Návrh řešení síťového provozu

Při návrhu a řešení tvarovaného síťového provozu je brán ohled na různé typy komunikace. Každý zákazník připojený k síti má od sítě jiné požadavky. Zákazníkem se myslí klientské zařízení. Tyto požadavky se dělí podle způsobu využívání přenosové rychlosti.

Finální síť má v sobě přes 900 klientských stanic. Každý klient je připojen bezdrátovým zařízením typu stanice na vysílací zařízení. Vysílacích stanic je v celkové bezdrátové síti přes 160. Přesné počty klientských a vysílacích stanic jsou v návrhu pouze orientační. Přesný počet se mění každý den, tudíž jsou počty zaokrouhleny. Všechny prvky v síti jsou zařízení od firmy Mikrotik. Tvarování sítě se provádí jak na serveru, tak na jednotlivých stanicích dle potřeby. V některých případech nestačí zajišťovat kvalitu služby pouze na hraničním serveru ale také na jednotlivých zařízeních v síti. Důvodem je například kontrola jednotlivých klientských stanic. Nejčastěji se toto provádí podle IP adresy. V některých případech klientské stanice dokážou vygenerovat velmi velký objem požadavků. Tyto požadavky se poté musí zpracovat. Aby nedošlo k zahlcení sítě, využívá se omezování provozu na zařízeních Mikrotik. Tyto zařízení mají možnost v sobě konfigurovat jednotlivé sítě. Jak monitorovat přenos od vysílače ke klientské stanici tak i samotnou klientskou síť. Pomocí dostupných funkcí v softwaru tak kontrolujeme v síti každého klienta a směrovač zvlášť. Výhodou je účinné řešení aktuálního problému díky možnostem softwaru od firmy Mikrotik.

Mimo hardware v síti, je základní požadavkem také dostupnost jednotlivých služeb. Je třeba zajistit jejich dostupnost a vhodnou rychlost zpracování. Pro tuto situaci jsem zvolil možnost serveru na jádře Linuxu. V tomto jádře je možnost využití mechanismu HTB. Tento mechanismus by měl vyhovět požadavkům na síť.

Navržená síť se skládá ze zařízení server s operačním systémem Debian verze 8 Jessie bez GUI (grafického prostředí), směrovače od firmy Mikrotik, přepínačů od firmy TP-Link a klientských stanic.

Hlavní server

Princip funkce hlavního serveru je zajistit potřebnou kvalitu služby QoS optimálně pro všechny klienty. Konkrétně se jedná o traffic shaping, který na serveru plní funkci tvarování provozu. Dále bude na serveru vedeno statické směrování, které se bude postupně rozšiřovat manuálním přidáváním nových podsítí (subnetů).

V poslední řadě bude na serveru firewall, který bude hlídat provoz v síti a bude také přiřazovat dle potřeby klientům uvnitř sítě veřejné IP adresy. Ve firewallu se budou také nastavovat priority pro určité typy komunikace. Zde bude komunikace označována a řazena do front. Mezi tyto typy komunikace může patřit například DNS nebo HTTP/HTTTPS.

Dále na serveru bude docházet k rozdělení provozu na dvě sekce, a to pomalý provoz a rychlý provoz. Rozdíl mezi těmito sekcemi je ten, že všichni klienti jsou z počátku automaticky

zařazeni do rychlejší sekce, která má nastavenou vyšší přenosovou rychlost než sekce s pomalým provozem. Důvodem je rozdělení přenosové rychlosti tak aby byla zachována rovnoměrná přenosová rychlost a priority mezi klienty. Pokud by nějaká klientská stanice generovala větší objem dat nebo požadavků, bude zařazen z rychlejší do pomalejší fronty. Příklad byl uveden na stránce 39, kde bylo vysvětleno stanovisko určitých klientů podle typu využívaných služeb.

Směrovače

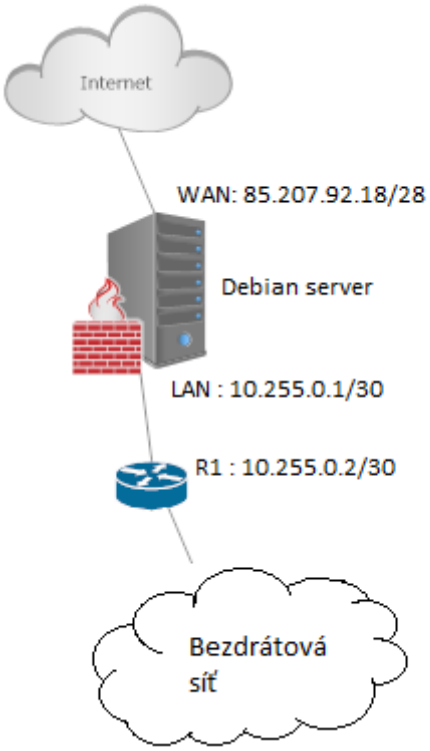
V našem návrhu sítě jsou dále použity směrovače od firmy Mikrotik. Ta vyrábí síťové zařízení jak pro ethernetové tak bezdrátové využití se softwarem postaveným na jádře Linux. V celé síti je těchto zařízení nespočet a každé zařízení označeno jako směrovač je typu Mikrotik. Tyto zařízení jsem si zvolil především pro jejich možnost konfigurovat různé vlastnosti sítě. Mezi tyto vlastnosti patří například rozdělení sítě. Zde je možnost rozdělovat sítě například podle ethernetových portů nebo VLAN a QoS. Tedy možnost určovat politiky a nastavovat pravidla či vytvářet fronty nebo omezení pro jednotlivé klientské stanice. Všechny zařízení v síti označené jako směrovače, AP nebo station jsou zařízení typu Mikrotik.

Klientské stanice

Každá klientská stanice má přijímací bezdrátové zařízení Mikrotik a dále je vedena jeho individuální domácí síť. Díky výhodám firmware je možné části QoS spravovat přímo u klienta v přijímacím zařízení. Což v konečném důsledku je pro návrh sítě velkou výhodou.

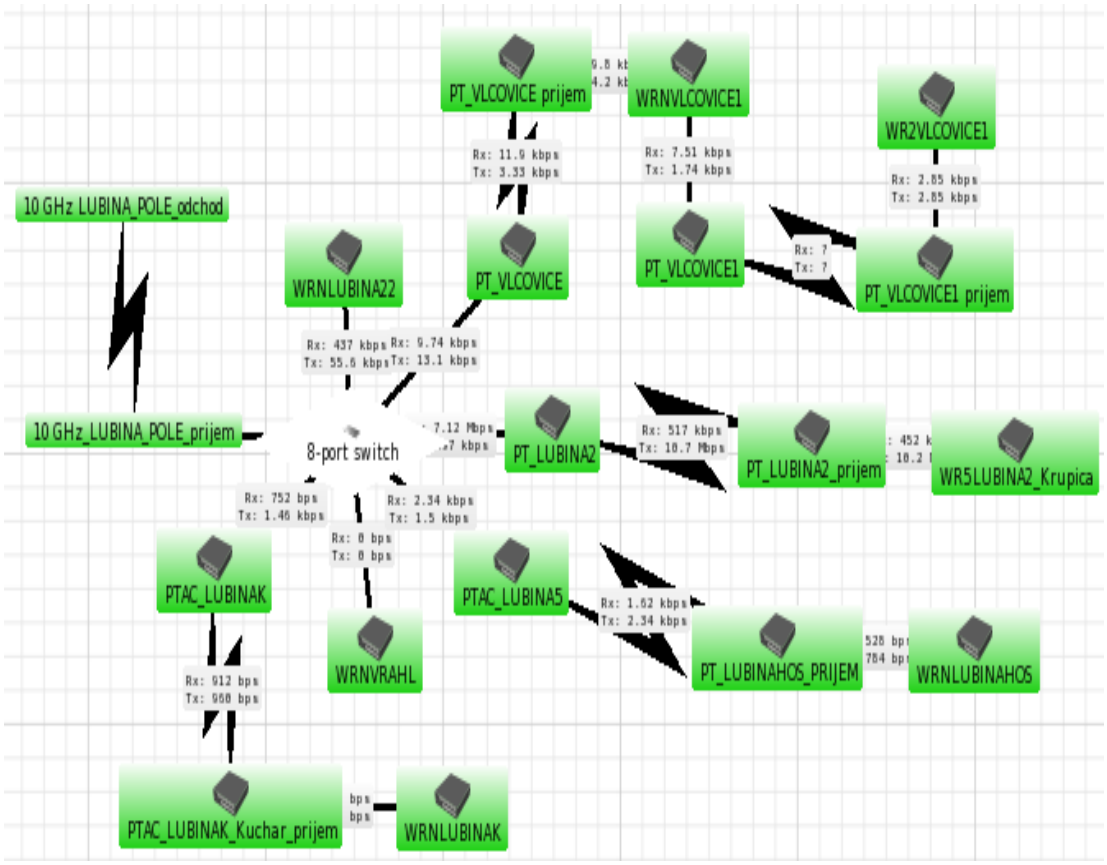
4.1 Kompletní návrh sítě

Kompletní návrh sítě je uveden na obrázku č 3.1. Tento návrh se skládá z připojení do Internetu, které je napojeno na serveru Debian. Náš server bude mít nastavenou WAN adresu 85.207.92.18/28 přiřazenou od poskytovatele přenosové rychlosti. Dále jsem si zvolil vnitřní adresu pro server. Pro vnitřní síť to je 10.255.0.1/30 propojenou ethernetem na první směrovač R1 v naší síti. Tento směrovač je první zařízení Mikrotik typu směrovač v síti. Na tomto zařízení je nastavena IP adresa 10.255.0.2/30. Na ethernetové rozhraní serveru je přivedena přenosová rychlost 1Gb/s. Tato přenosová rychlost je zpracovávána na serveru. Je na ní také přidělený určitý rozsah veřejných IP adres pro potřeby klientských stanic.

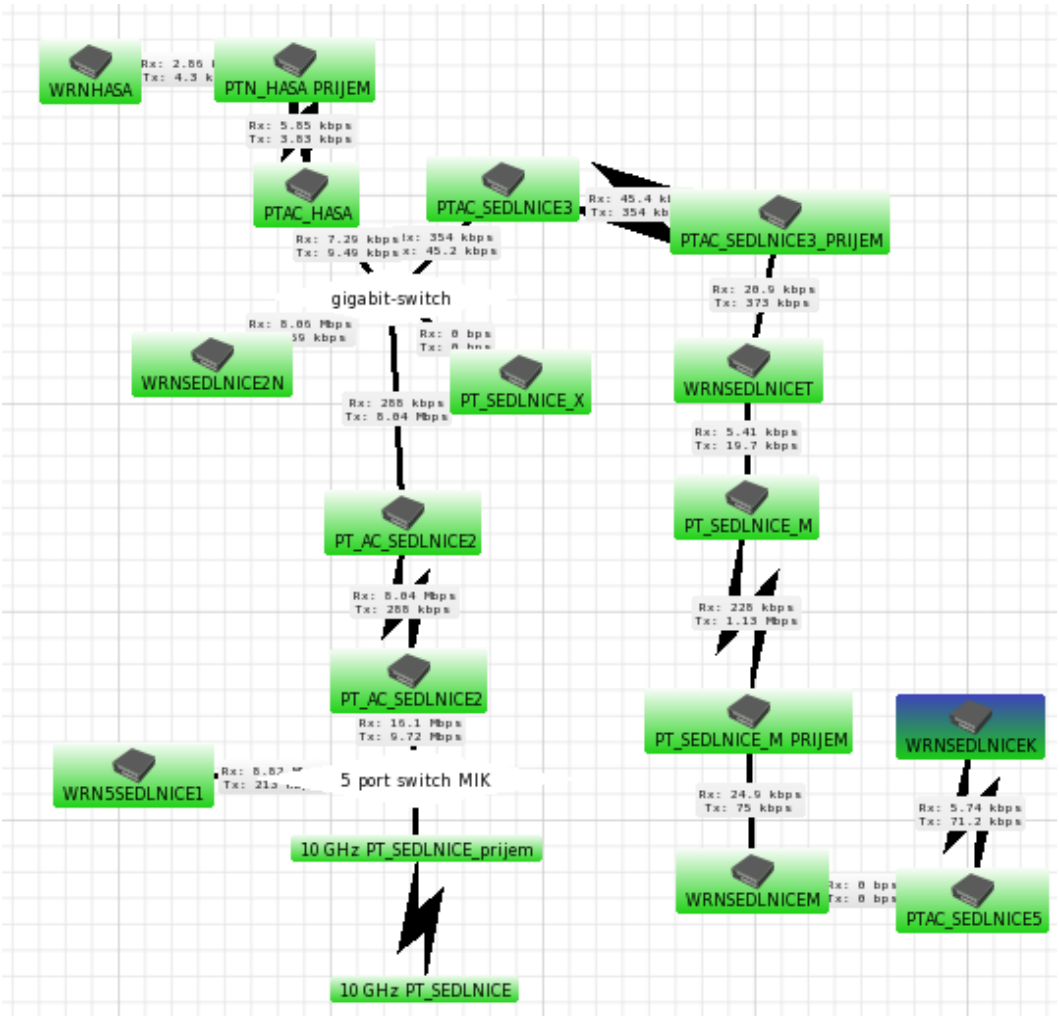


Obrázek 4.1: Schéma zapojení

Celková bezdrátová síť je příliš velká a obsahuje velké množství zařízení. Je zde několik stovek přepínačů a směrovačů. Pro návrh sítě je velmi nepřehledné popisovat síť jako celek. Z toho důvodu je celá tato síť nazvána jako bezdrátová síť na obrázku č .4.1. V bezdrátové síti jsou vysílače a přijímače postaveny do hvězdicového zapojení. Pro představení uvedu část zapojení na obrázku č. 4.2 a obrázku č. 4.3 níže jsou představeny způsoby zapojení bezdrátové sítě. Na obrázku č. 4.2 je schéma pro zapojení v oblasti Lubina, které spadá pod oblast Kopřivnice. Na obrázku č. 4.3 je schéma zapojení pro oblast Sedlnice. Na obrázku je také zmíněný první směrovač v síti R1. Tento směrovač je první zařízení v interní síti. Jedná se konkrétně o Mikrotik cloud core router CCR1009. Tento směrovač propojuje všechny lokality v interní síti. Směrovač R1 je první a jediný prvek, který celou bezdrátovou interní síť přeposílá na zpracování do serveru.



Obrázek 4.2: Příklad části sítě lokalita Lubina



Obrázek 4.2: Příklad části sítě lokalita Sedlnice

Výše zmíněné sítě jsou pořízeny v programu The Dude. Jde o software od firmy Mikrotik pro kreslení topologie a monitoring zařízení. Ostrost jednotlivých zařízení není podstatná, jelikož výše zmíněné obrázky slouží pouze jako příklad pro bezdrátovou síť. Těchto zařízení je ve vnitřní bezdrátové síti několik stovek. Na obrázcích výše můžeme vidět dva typy propojení mezi zařízeními. První typ je styl rovné čáry, který naznačuje ethernetové propojení mezi zařízeními. Druhý typ je formou blesku, který naznačuje bezdrátové spojení mezi zařízeními. Tento typ označení je typický pro program The Dude. V ostatních sítích může být zobrazení pro bezdrátovou komunikaci nebo ethernetovou komunikací jiné.

5 Realizace navržené sítě

Praktická část se zabývá konkrétním řešením pro naši síť. Přesněji jak celý server funguje, a probereme si i příklady z nastaveného QoS a firewallu. Realizace je provedená na zátěži o více než 160 zařízení v síti typu směrovač, a na více než 900 klientských stanic. Toto všechno zpracovává server, na který je přivedená garantovaná přenosová rychlost 1 Gb/s poskytovatelem tzv. konektivity.

Typ a složení serveru:

Obrázek č.5.1- Příklad serveru použité pro naší síť

Jedná se o server SUPERMICRO 1U. Server obsahuje dva procesory Intel s paticí LGA2011-3, registrované paměti ECC RAM o velikosti 16 GB, dva disky o velikosti 1 TB, dvě síťově 10-gigabitové karty a rozhraní pro IPMI.

Dále je důležitý první směrovač, který zpracovává celou bezdrátovou síť. Tento směrovač spojuje celou bezdrátovou síť. Typ směrovače níže:

Router R1 – Mikrotik cloud core router CCR1009

5.1 Základní realizace

Na serveru jsou tedy 3 síťové karty. První síťová karta bude sloužit pro konfiguraci a je označena ETH1. Zde na tento port ETH1, který bude naším WAN rozhraním je přivedená garantovaná přenosová rychlost 1 Gb/s. Druhá síťová karta slouží pro spojení s interní bezdrátovou sítí LAN a je označena jako ETH2. Třetí síťová karta je pro obsluhu a na této kartě bude napojeno IPMI rozhraní. Toto rozhraní je označeno jako ETH0. Na WAN síť byla přidělena veřejná IP adresa 85.207.92.16/28. Na LAN síť byla nastavena IP adresa 10.255.0.1/30. Na konfigurační port je nastavena adresa 85.207.92.15/24. Do našeho serveru je přivedena garantovaná přenosová rychlost 1 Gb/s.

Než se pustíme do konfigurace QoS ujasníme si, jak server bude fungovat z pohledu směrování. Jak bylo zmíněno, tak na serveru je použito statické směrování. Aby v naší síti pakety putovaly správnými cestami, je třeba správně konfigurovat směrovací tabulku. Konkrétně pro náš server tabulka vypadá následovně:

Adresát	Brána	Maska	Užt Rozhraní
0.0.0.0	85.207.92.17	0.0.0.0	eth1
85.207.92.16	0.0.0.0	255.255.255.240	eth1
10.0.0.1	0.0.0.0	255.255.255.252	eth2
10.122.5.0	10.0.0.2	255.255.255.0	eth2
85.207.92.15	85.207.92.17	255.255.255.240	eth0

Tato směrovací tabulka popisuje podsítě v naší síti. První řádek znamená, že veškerý příchozí datový tok bude předaný bráně 85.207.92.17/28 a propuštěn ven ze sítě. Maska je odvozena od prefixu u každé IP adresy. Přehled tabulky dle CIDR naleznete ve zdrojích v odkazu na konci diplomové práce. Při použití IP adresace je třeba oddělit si rozsahy adres použitelné pro klienty a od adres používaný na jednotlivých směrovačích. Klientské adresy budou mít vždy prefix /24 a adresy mezi jednotlivými směrovači budou mít prefix dle potřeby. Brána 10.0.0.1/30 představuje IP adresu prvního směrovače v propojení mezi naším serverem a směrovačem z prvního návrhu sítě. V návrhu vidíme, že dále se síť nevětví a dále následují připojení klienti. Z toho důvodu je v řádku 4 vytvořen rozhas IP adres sítě pro klienty 10.122.5.0/24. Tato směrovací tabulka není kompletní. Slouží pro vzorový příklad. Kompletní směrovací tabulka má v sobě stovky podsítí, a z pohledu její velikosti není vhodná pro demonstraci. Nyní přejdeme to vytváření pravidel ve firewallu.

Představíme si standartní nastavení serveru bez QoS. Jelikož náš server neobsahuje pouze firewall a překlad adres ale také funguje jako směrovač, je zapotřebí pomocí iptables vytvořit pravidlo pro průchod veškeré komunikace mezi rozhraními ETH1 a ETH2. Prvním krokem ale než začneme vytvářet pravidla je ujistit se, že aktuálně nejsou žádné pravidla vytvořeny. To zajistíme následujícími příkazy.

```
iptables -F
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -F -t nat
iptables -F -t mangle
```

První pravidlo smaže všechny nastavené předchozí pravidla. Druhé pravidlo má za funkci zahazovat veškerý provoz, který přichází na vstupní rozhraní. Třetí pravidlo zahazuje veškerý provoz snažící se dostat mezi vnitřní a vnější sítí. Čtvrté pravidlo vyčistí veškeré předchozí nastavení v NAT (Network Address Translation). Páté pravidlo vyčistí veškeré předchozí nastavení v mangle.

Po přípravě a smazání veškerého předchozího nastavení se můžeme vrhnout na vytváření pravidel. Začneme od pravidel pro předávání pakety mezi rozhraními a jejich povolení vkročit do sítě.

```
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -i $dev_inet -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Jako první je třeba nastavit pravidlo, aby vše co přijde na nás loopback prošlo. To zaručí první pravidlo. Dále chceme, aby veškerý příchozí traffic na vstup ETH2 z naší vnitřní sítě prošel, to zaručí druhé pravidlo. Jelikož používáním firewallu využíváme součást jádra netfilter, což je firewall je potřeba ještě k plné funkčnosti zajistit aby se propustil provoz mezi navázanými spojeními při vstupu. To zajistí třetí pravidlo.

Toto všechno ale ještě nestačí. Ještě je třeba nadefinovat pravidla pro propuštění mezi našimi WAN a LAN sítěmi. K tomu použijeme následující dvě pravidla.

```
iptables -A FORWARD -i eth1 -o $dev_inet -m state --state
RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i $dev_inet -o eth1 -j ACCEPT
```

První pravidlo zajistí, že příchozí spojení na ETH1 se přepustí na ETH2. Druhé pravidlo zajistí, že vše co přijde na ETH2 se přepustí na ETH1. Těmito všemi pravidly jsme si nastavili základní politiku. Tato politika říká, že veškerý provoz se navzájem propustí mezi rozhraními ETH1 a ETH2, a vše co vstoupí na loopback se taktéž propustí.

Dále je třeba nastavit NAT mezi našimi rozhraními WAN a LAN. Toho dosáhneme použitím následujícího příkazu:

```
iptables -t nat -A POSTROUTING -o eth1 –j SNAT 85.207.92.16/28
```

Toto pravidlo nám zaručí překlad veškerého příchozí provozu na adresu 85.207.92.16/28. Povšimněme si, že jsem nepoužil pravidlo pro MASQUERADE ale místo toho jsem použil pravidlo SNAT. Udělal jsem to, ze dvou důvodů. První je, že nepotřebujeme použít MASQUERADE z důvodu, že na výstupním rozhraní WAN máme nastavenou pevnou IP adresu, která se měnit nebude. Způsob pro překlad způsobem MASQUERADE se používá, pokud se dynamicky mění IP adresa na výchozím rozhraní WAN. Druhým důvodem proč jsem využil SNAT je, že SNAT sleduje veškeré připojení. Pokud se stane situace, že by se rozhraní WAN přerušilo a znovu navázalo, tak spojení nezmizí a zůstanou zachované. Což při použití MASQUERADE se veškeré připojení smaže.

Pravidla pro průchod mezi rozhraními máme připravené nyní je potřeba zapnout přepínání mezi interface WAN a LAN. To uděláme zápisem stavu 1 místo 0 do ip_forward viz níže.

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Nyní je třeba nejdůležitější parametr a to propuštění paketů ICMP ven ze sítě. Také potřebujeme, abychom mohli přistupovat vzdáleně na náš server povolit příchozí provoz SSH.

```
iptables -A INPUT -p icmp -j ACCEPT #ICMP
iptables -A INPUT -p tcp --dport 22 -j ACCEPT #SSH
```

První pravidlo veškeré příchozí pakety ICMP okamžité propustí ven ze sítě. Druhé pravidlo zajišťuje, že cílový port 22 (SSH) bude také propuštěn.

Nyní máme základní pravidla nastaveny a můžeme přejít k nastavení tříd jednotlivých tříd.

```
tc qdisc del dev $dev_inet root

tc qdisc add dev $dev_inet root handle 1: htb default 10

tc class add dev $dev_inet parent 1:0 classid 1:1 htb rate
1000Mbit ceil 1000Mbit
```

Následující příkazy v prvním bodě smažou veškeré vytvořené pravidla na rozhraní ETH2. Dále si nastavíme ETH2 jako root interface. Poté vytvoříme na ETH2 třídu, která bude mít určité hodnoty rate a ceil nastaveny na 1000 Mb/s což je přivedena přenosová rychlost na náš server na rozhraní WAN.

Třídami výše jsme sice omezili download linku klientům, ale musíme myslet i na upload. Pokud by se neomezil upload, mohlo by dojít k situaci k zhroucení sítě. Pokud by některý z klientů například nahrával větší objem dat, tak by při tom bez kontroly provozu mohl datově spotřebovat veškerou přenosovou rychlost. To by mělo za následek zvýšení latence při průchodu přes server ven nebo v nejhorším případě přerušení spojení s venkovní sítí. Abychom takové situaci předešli, musíme provoz tvarovat. Vhodným tvarováním provozu dosáhneme ochrany před takovouto situací. Dosáhneme toho tím, že nejen na ETH2 vytvoříme třídu, ale vytvoříme třídu také pro ETH1 kde budeme kontrolovat upload.

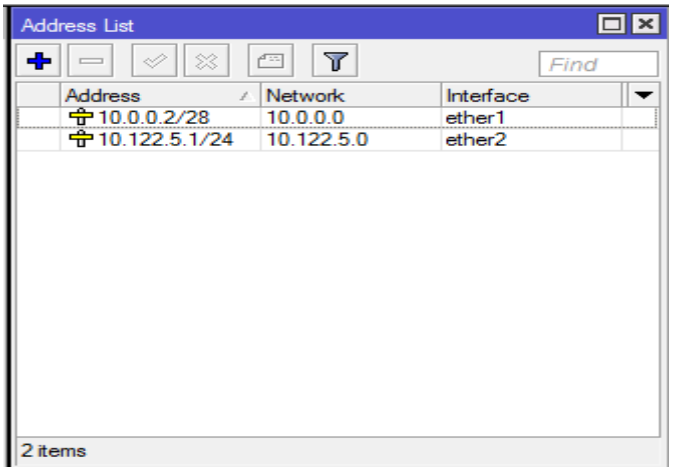
```
tc qdisc del dev eth1 root

tc qdisc add dev eth1 root handle 2: htb default 10

tc class add dev eth1 parent 2:0 classid 2:2 htb rate 1000Mbit
ceil 1000Mbit
```

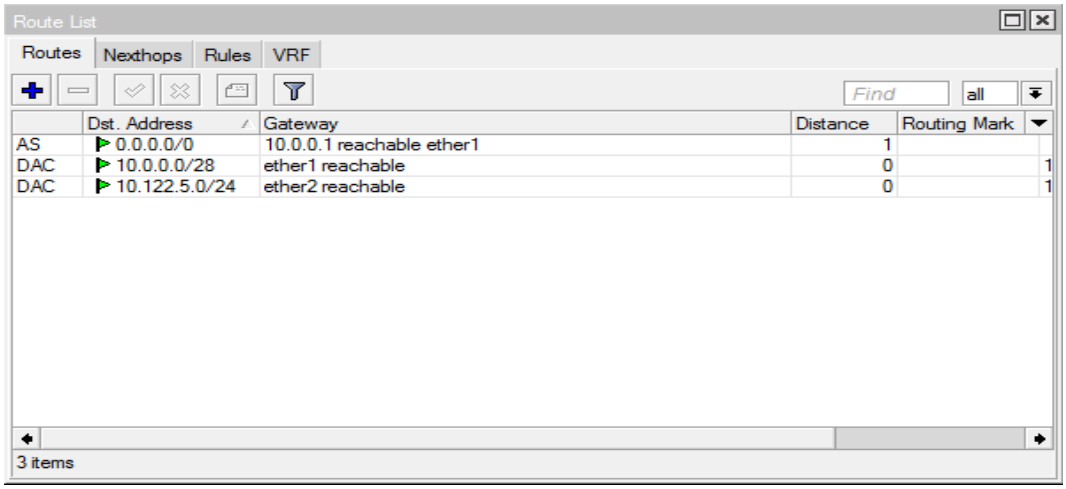
První řádek vymaže předchozí konfiguraci ETH1. Druhý vytvoří root třídu. Následně vytvoříme třetí třídu s rychlostí uploadu maximálně 1000Mbit.

Po nastavení základních parametrů máme připravenou síť pro chod. Pouze ale se základní konfigurací, kde nám ještě chybí rozpoznávání a rozdělování provozu dle služeb. Tuto konfiguraci budeme nastavovat později.



Obrázek 5.2: Příklad ip addres ze směrovače R1

Nastavení směrovací tabulky v R1.



Obrázek 5.3: Příklad směrovací tabulky ze směrovače R1

Na obrázku č. 5.2 je uveden první směrovač R1 v naší síti. Zde je mu přidělená IP adresa 10.0.0.2/30. Tento subnet slouží pro propojení serveru a směrovače R1. Na obrázku č. 5.3 je uvedena směrovací tabulka, kde je přidána v prvním řádku adresa výchozí brány. V tomto případě je výchozí brána náš server a má adresu 10.0.0.1/30. Obrázky slouží jako příklad pro prostředí Mikrotik a základní nastavení prvního směrovače v síti. V praktické části je na směrovači R1 podstatně větší počet IP adres a také směrovací tabulka je velmi rozsáhlá. V tak rozšířené síti by příklady použitých IP adres a směrovací tabulky byly velmi nepřehledné. Z toho důvodu je v této práci uveden jednoduchý příklad na obrázcích výše.

5.2 Podrobnější doplnění konfigurace

Po nastavení základních pravidel pro funkční chod sítě, musíme konfiguraci upřesnit a dodělat. Bez tohoto upřesnění by základní konfigurace nestačila pro dodržení požadavků na funkci sítě. V podrobnější konfiguraci budeme přiřazovat určitou část přenosové rychlosti pro jednotlivé služby. Při menším množství klientských stanic není třeba vytvářet složité konfigurace. V menším počtu klientských stanic si vystačíte i se základní konfigurací. Jak bylo zmíněno na začátku práce, v naší verzi máme stovky klientských stanic. Vytvářet tedy pro každou klientskou stanici vlastní třídu by nemělo smysl. Třídy by v takovém množství byly nepřehledné. Lepší volbou je zvolit rozdělení provozu dle služeb. Tyto služby v provozu označovat ve firewallu a dle jejich označení je zpracovávat. K tomu využijeme metodu HTB a budeme provoz třídit a zpracovávat podle priorit. Musíme si první určit priority. Samozřejmě jako první budeme chtít zpracovávat služby citlivé na čas. Mezi takové služby můžeme zařadit například DNS, HTTP nebo ICMP pakety. Tyto pakety tedy budeme odesílat z naší sítě přednostně. Zařadíme je tedy do třídy s nejvyšší prioritou. Třídy s nejnižší prioritou jsou zpracovávány jako poslední. V HTB pakety označené nižším číslem u zadané priority, jsou zpracovávány dříve než pakety ostatních priorit. Dále budeme rozeznávat také komunikaci SMTP, POP3 a IMAP. Ostatní provoz v síti bude řadit zvlášť do třídy. Nakonec se vytvoří ještě třída pro klienty, kteří generují, přeposílají nebo stahují větší objem dat a tím zahlcují síť. V praxi se běžné setkávám se situací, kdy jedna klientská stanice dokáže vyprodukovat množství dat, dost veliké na to zahltit síť. A to nejen svou vlastní ale i také naší bezdrátovou síť. V bezdrátové síti to má zásadní vliv. Pokud se taková situace stane, zvýší latence ostatním klientským stanicím. V krajním případě dojde s rozpojení bezdrátového spoje. Z toho důvodu je do sítě ještě přidáno pravidlo do firewallu na serveru. Níže vypsané pravidlo takové klientské stanice odhalí a zařadí je do pomalejší třídy. V této třídě jsou klientské stanice umístěny, dokud nejsou ručně opět zařazení zpátky do fronty původní. Využití této možnosti je nezbytné pro stabilnější bezdrátové sítě. Bez tohoto nastavení by mohlo docházet k náhodným neočekáváným nestabilit bezdrátové sítě. Což by mělo vliv i na ostatní klientské stanice a ne pouze na tu, která tuto situaci způsobila. Příklad příkazu níže:

```
iptables –t mangle –A FORWARD –m connbytes –p tcp –conbytes
600000 –connbytes-dir –connbytes-mode bytes –j mark_stahovac
```

Toto pravidlo zavádí možnost sledovat TCP provoz jednotlivých klientských stanic. Pokud některá klientská využívá TCP spojení déle jak 60 vteřin je zařazen do třídy mark_stahovac. Tato třída označuje klientské stanice, které využívají spojení déle než 60 vteřin a pravděpodobně vytvářejí větší objem dat. Takové chování se dá přirovnat službám využívající větší množství spojení s více místy na světe. Na tyto místa nadále maximální možnou přenosovou rychlostí posílají nebo stahují objemy dat. Tím využívají přenosovou rychlost potřebnou pro naše služby. My tuto rychlost však potřebujeme využít primárně na služby, které

přenosovou rychlost nutně potřebují. Rozhodl jsem se využít přenosovou rychlost do tříd následovně.

První třída bude tvořit 55% z celkové přenosové rychlosti. Tedy dohromady 550 Mb/s. V této frontě budou přiřazeny pakety s vysokou prioritou a to služby typu HTTP, HTTPS,SMTP,POP3 a IMAP.

Druhá fronta bude tvořit 20% z celkové přenosové rychlosti. Tuto frontu budou tvořit DNS a ICMP pakety. Na tyto služby bude využita rychlost 200 Mb/s.

Třetí fronta bude tvořit 10% z celkové přenosové rychlosti. Do této fronty bude řazen zbytek provozu. Pro tuto frontu je vyhrazena přenosová rychlost 100 Mb/s.

Čtvrtá fronta označena jako mark stahovač bude mít přiřazenou přenosovou rychlost 100 Mb/s. Tato fronta bude mít tedy přenosovou rychlost o velikosti 10% kapacity celkové přenosové rychlosti. Dostupných bude 100 Mb/s pro klientské stanice, kteří jsou zařazeny do pomalejší fronty. U této fronty bude nejvyšší počet zahozených paketů. Důvodem je dlouhé a nepřetržité TCP spojení. Celková přenosová rychlost činí 1000 Mb/s. Po součtu všech čtyř front nám ale 50 Mb/s chybí. Důvodem je záloha přenosové rychlosti určená pro bursty a hardware.

Vytvoříme si tedy čtyři třídy. První třídu budeme nazývat rychlá fronta. V této frontě budou pakety s vysokou prioritou. Tyto pakety budou mít nejvyšší prioritu, jelikož jsou náchylné na dobu odeslání. Jejich priorita je tedy nejvyšší a to 0. Dále si nadefinujeme třídu pro ICMP a DNS. Této třídě dáme prioritu 1 a přidělíme ji vlastní přenosovou rychlost 200 Mb/s. Třetí třídu pro ostatní provoz nastavíme na prioritu 1 a přidělíme ji přenosovou rychlost 100 Mb/s. U této třídy je předpoklad, že zde nebude příliš velký objem dat. Z toho důvodu jsem zvolil 100 Mb/s. Poslední čtvrtá třída vyhrazená pro větší objemné data, má přiřazenou přenosovou rychlost 100 Mb/s.

#rychlá fronta SIP,SMTP,HTTP,HTTPS,POP3,IMAP

```
tc class add dev $dev_inet parent 1:0 classid 1:10 htb burst
10000 cburst 10000 rate $(( $bw_fast ))Mbit ceil $(( $bw_fast
))kbit prio 0
```

#fronta pro ICMP a DNS

```
tc class add dev #dev_inet parent 1:0 classid 1:11 htb burst
10000 cburst 10000 rate $(( $bw_web ))Mbit ceil $(( $bw_web
))kbit prio 1
```

#Fronta pro ostatní provoz

```
tc class add dev #dev_inet parent 1:0 classid 1:12 htb burst
10000 cburst 10000 rate $(( $bw_other ))Mbit ceil $(( $bw_other
))kbit prio 1
```

#Fronta pro stahovače

```
tc class add dev $dev_inet parent 1:0 classid 1:13 htb burst
10000 cburst 10000 rate $(( $bw_stahovac ))Mbit ceil $((
$bw_stahovac
))kbit prio 1
```

Po vytvoření jednotlivých tříd musíme nějakým způsobem přiřazovat pakety určitých služeb do tříd, kam patří. K tomu využijeme vlastnosti Linuxového jádra. Započneme jako první s třídou určenou pro objemné data nazvanou mark_stahovac. Na to využijeme informace uložené v TOS a v DSCP. První je třeba přiřadit označení mark_stahovac nějakou hodnotu v DSCP poli.

Tímto pravidlem vytvoříme označení pro mark_stahovac

```
iptables -t mangle -N mark_stahovac
```

Dále je třeba dávat si pozor, zda už neexistuje předchozí mark, zjistíme to příkazem

```
iptables -t mangle -A PREROUTING -j CONNMARK --restore-mark
```

Pokud tedy paket nemá marku, označíme ho, že patří do fronty mark_stahovac..

```
iptables -t mangle -A PREROUTING -m mark --mark 0 -j
mark_stahovac
```

Po označení musíme paketu přidat přesnou marku a DSCP hodnotu.

```
iptables -t mangle -A mark_stahovac -m dscp --dscp 0x02 -j MARK
--set-mark 0x08
```

Když máme paket označen, můžeme jej přiřadit jednotlivé třídě. V našem případě přiřadíme interface $DEV filter, který bude zachytávat pakety s TOS 0xb8 což je označení pro dscp=0x02.

```
tc filter add dev $dev_inet parent 1:0 protocol ip prio 0 u32
match ip tos 0x08 0xff flowid 1:10
```

V konečném bodě marku uložíme.

```
iptables -t mangle -A mark_stahovac -j CONNMARK --save-mark
```

Tímto máme vyřešenou třídu pro mark_stahovac. Dále pokročíme na třídu pro SIP,SMTP,HTTP,HTTPS,POP3 a IMAP.

První nastavíme pro výše zmíněné služby hodnotu v DSCP:

```
iptables -t mangle -A FORWARD -p udp -m multiport -dports
5060,80,443,143,993,995,587 -m dscp --dscp 0x01 -j DSCP --set-
dscp 0x01
```

Poté co označíme pakety podle hodnoty v DSCP je můžeme přiřadit určité třídě pomocí filtru.

```
tc filter add dev $(dev_inet)s protocol  ip parent 1: pref 10
u32 match u8 0x04 0xfc at 1 flowid 1:11
```

Jako poslední nám schází zařadit pakety DNS do určité třídy. Toho dosáhneme následovně.

Nastavíme pro službu DNS hodnotu v DSCP:

```
iptables -t mangle -A FORWARD -p udp --sport 53 -m dscp --dscp
0x00 -j DSCP --set-dscp 0x00
```

Poté ji opět pomocí filtru přiřadíme do potřebné třídy :

```
tc filter add dev $(dev_inet) protocol ip parent 1: pref 10 u32
match u8 0x00 0xfc at 1 flowid 1:10
```

Nyní máme vytvořené třídy. Označujeme pakety dle služeb a přiřazujeme i do patřičných tříd. Poslední co musíme udělat je navázat ještě na každou třídu frontovou disciplínu. To uděláme následovně:

```
tc qdisc add dev $dev_inet parent 1:10 handle 10:0 sfq perturb
10

tc qdisc add dev $dev_inet parent 1:11 handle 11:0 sfq perturb
10

tc qdisc add dev $dev_inet parent 1:12 handle 12:0 sfq perturb
10

tc qdisc add dev $dev_inet parent 1:13 handle 13:0 sfq perturb
10
```

Po vytvoření všech pravidel pro kvalitu služby je třeba nastavit základní parametry pro ochranu naší sítě. V praxi je nejběžnější využívat zranitelnost využívaných zařízení. V našem případě je velmi časté, že z venkovní sítě přichází dotazy na zařízení Mikrotik. A to především na porty, které má Mikrotik defaultně nastaveny v sobě. Mezi tyto porty patří port 8291. Aby nedocházelo k útokům na zařízení Mikrotik přes port 8291, rozhodl jsem se jej pro přístup z vnější sítě do vnitřní sítě zakázat.

Pravidlo pro zakázání přístupu na Mikrotik zvenčí:

```
Iptables -A INPUT -i eth1 -p tcp --dport 8291 -j DROP
```

Pro přístup na zařízení Mikrotik lze využít také proxi. Samotné směrovače podporují připojení tohoto typu. Aby nedocházelo k nežádanému připojování, rozhodl jsem se blokovat na serveru port 1080.

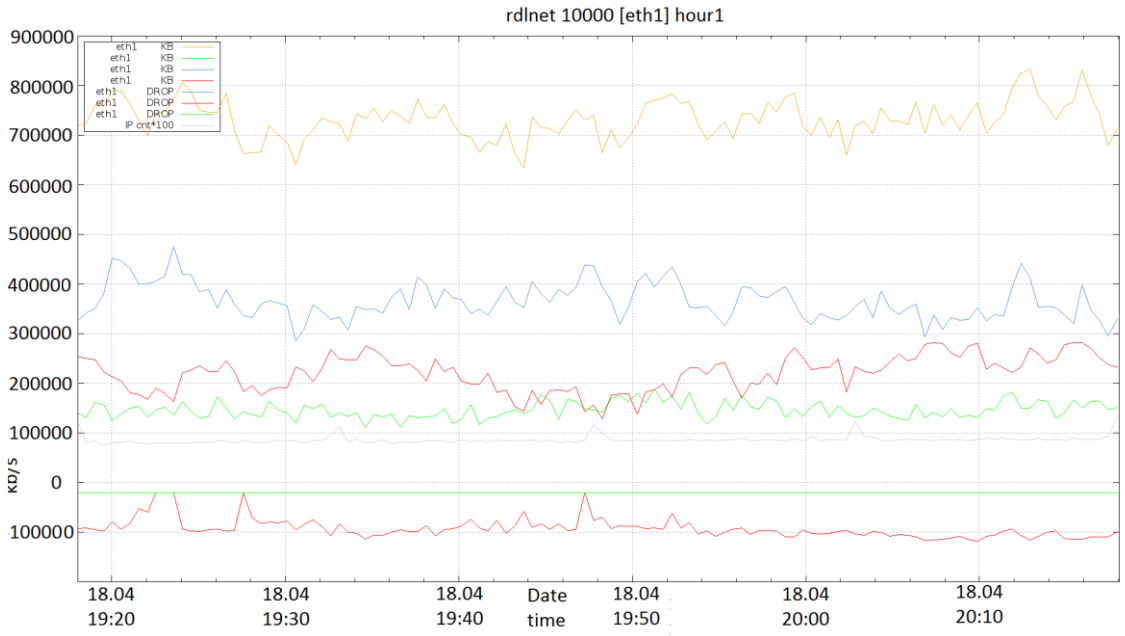
```
iptables -A FORWARD -i eth1 -p tcp --dport 1080 -j REJECT
```

Dále jsou docela častým jevem útoky z venkovní sítě na port 53 (DNS). Ve směru toku dat do naší sítě to může znamenat bezpečnostní riziko. Pro ochrany vnitřní sítě jsem se rozhodl dotazy do vnitřní sítě na port 53 zakázat. A to pomocí příkazu níže.

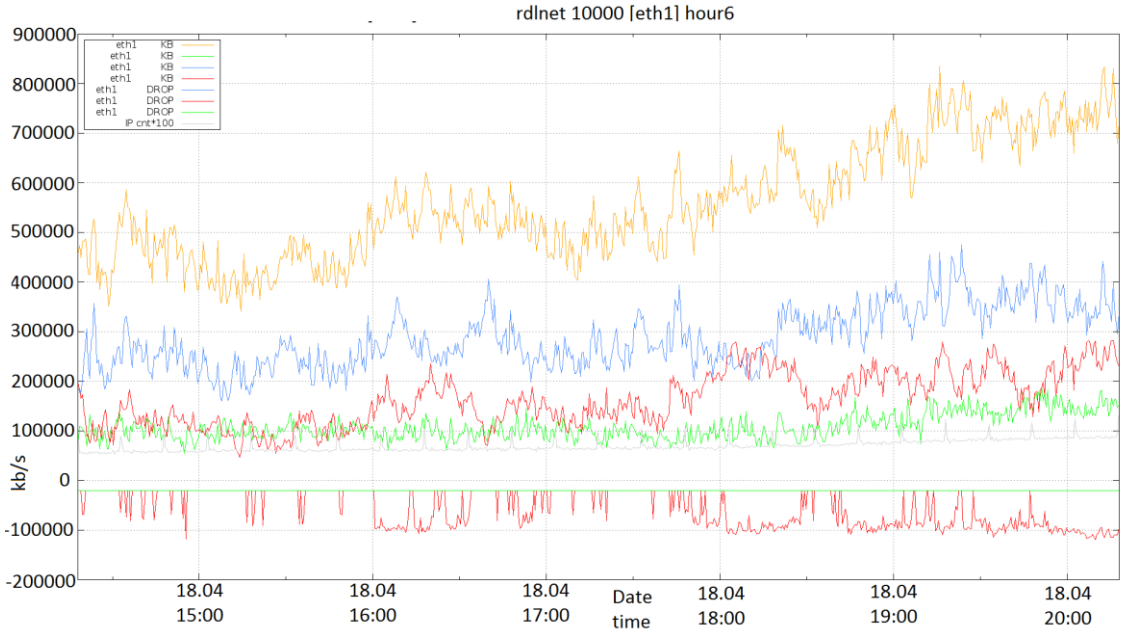
```
iptables -A INPUT -i eth1 -p tcp --dport 53 -j DROP
iptables -A INPUT -i eth1 -p udp --dport 53 -j DROP
```

6 Vyhodnocení

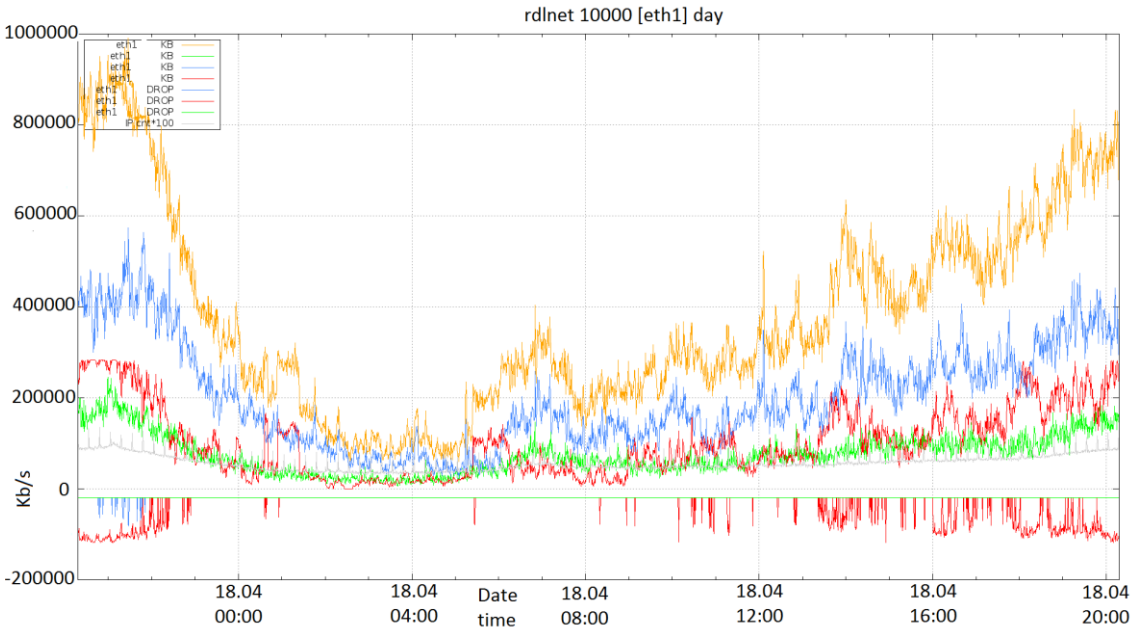
Výsledkem provozu jsou čtyři následující grafy. Na tyto grafy byly použity informace za dobu provozu uložené v souboru. Historii uložených dat se uchovává nejdéle na dobu jednoho měsíce. Na těchto grafech lze vidět námi nadefinované třídy. Tyto třídy mají svou přidělenou přenosovou rychlost. Grafy potvrzují, že naše konfigurace funguje správně. Výsledkem jsou grafy vykreslené v aplikaci gnuplot. Výsledek je rozdělen na 4 grafy. Tyto čtyři grafy jsou vyhodnocením aktivního serveru po dobu 1 hodina, 6 hodin, 1 den a 1 týden.



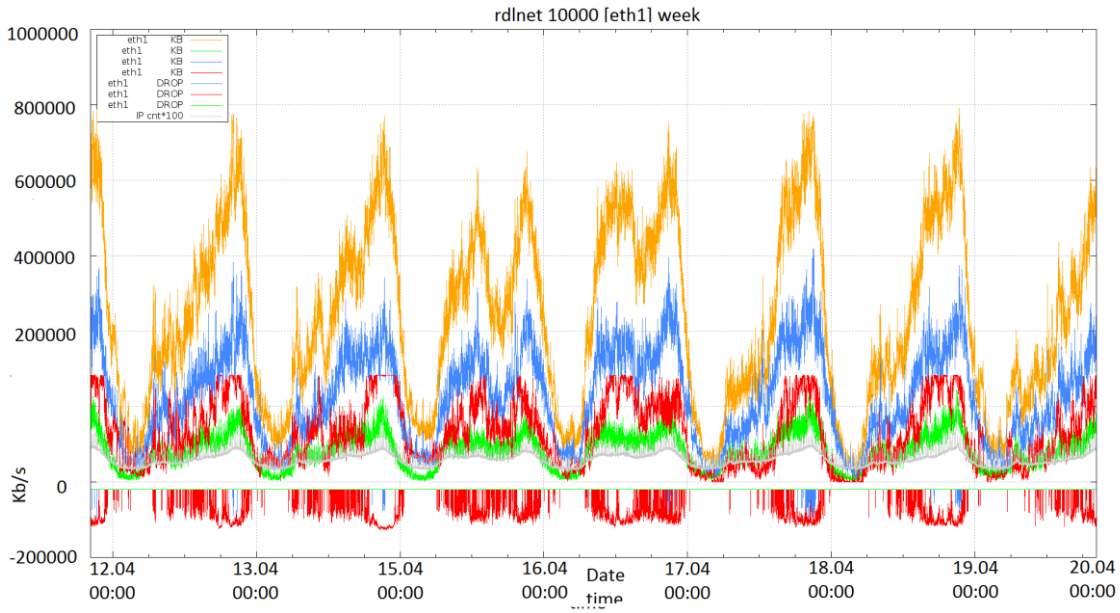
Obrázek 6.1: Výsledný graf QoS 1 hodin



Obrázek 6.2: Výsledný graf QoS 6 hodin

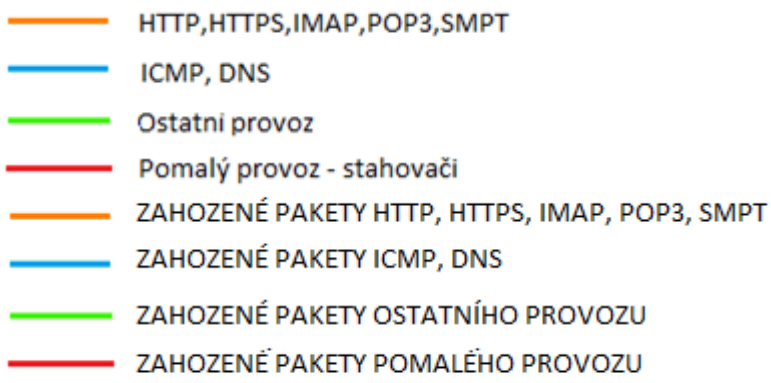


Obrázek 6.3: Výsledný graf QoS 1den



Obrázek 6.4: Výsledný graf QoS 1 týden

Legenda grafu:



Z grafů je patrný vývoj využití služeb v čase 1 hodiny, 6 hodin, 1 dne a 1 týdne. Z grafu vyplývá, že nejvíce využívanou službou v síti jsou webové služby HTTP/HTTPS a emailové služby IMAP,POP3 a SMTP. Tyto služby tvoří většinu paketů v síti. V porovnání s ostatními třídami, které jsme si nadefinovali, je tato třída využívána objemově jako zbylé tři třídy dohromady. Druhá nejvyužívanější třída je pro DNS a ICMP pakety. Drobnou odchylkou v druhé třídě pro datové pakety DNS a ICMP, může být systém pro informování stavu sítě. V síti máme navíc funkční SMS bránu, která se pomocí ICMP zpráv dotazuje na přístupnost jednotlivých směrovačů. Pokud tento směrovač neodpovídá zašle zařízení SMS o stavu směrovače v podobě DOWN. Pokud zařízení odpoví a opět funguje, zašle se zpráva v podobě UP. Dále ještě podle latence se zasílají SMS zprávy zda je bezdrátový spoj v pořádku. Pokud

totiž mezi vysílačem a přijímačem dochází k vysokým latencím, zašle se zpráva buď SLOW nebo FAST. Zpráva SLOW znamená vysokou odezvu mezi vysílačem a přijímačem. Příčina může být buďto vysoké rušení v pásmu, na kterém vysílač vysílá nebo vysoký datový tok od klientské stanice. Místy se stává, že některé klientské stanice se snaží využít vyšší přenosovou rychlost, než dokáže bezdrátový spoj zvládnout. Kvůli využití vyšší přenosové rychlosti naroste latence a to může způsobit zprávu SLOW. Třetí třída pro ostatní provoz není zcela využita. Je zde mírný datový tok. Odhad pro přenosovou rychlost vyhrazenou této třídě byl správný. Poslední čtvrtá třída označená červenou barvou je pro nás zásadnější. V této frontě jsou naši dříve pojmenovaní stahovači. Tito klienti mají vyhrazenou malou část přenosové rychlosti. Nejvyšší možnou vzhledem k využití ostatních služeb.

V grafu je důležité povšimnou si nulové hodnoty. Lze si všimnout, že některé barvy se vyskytují pod osou označenou 0. Barvy vyskytující se pod přímkou 0 jsou označením pro zahozené pakety. Tyto pakety jsou z důvodu využití HTB pro tvarování provozu zahozeny. Výše jsme si vysvětlovali jak systém HTB funguje. Lze vidět, že nejčastější zahozené pakety jsou určené pro objemné datové toky. Zde dochází k tak vysokým nárokům, že tato třída zahazuje pakety velmi často. Při odesílání nebo spíše při stahování většina klientských stanic nebere ohledy na definované třídy. Snaží se tím překročit nadefinovanou přenosovou rychlost a tím dochází k zahazování paketů. Dokonce i v případě, kdy je paket zahozen, tak generování datových toků nepřestává a pokračuje dále. Zahazováním TCP zároveň částečně snižujeme přenosovou rychlost. To z důvodu, že pokud je TCP paket zahozen, pošle zprávu o nedoručení TCP paketu a opět si o paket požádá. Zahozením více TCP paketů tedy prodlužujeme čas potřebný k splnění kompletního přenosu a tím snižujeme přenosovou rychlost. Ovšem v některých situacích nastane situace, že není další přenosová rychlost služby k dispozici a začnou se zahazovat i pakety jiných služeb. Můžeme si povšimnout, že pod nulovou osou jsou občas vidět i jiné barvy než červená. Na obrázcích č. 6.1 a 6.2, které jsou z krátkodobého provozu nelze moc dobře rozpoznat zahazování paketů jiných služeb. Nejlépe tato situace jde rozeznat z obrázků č. 6.3 a 6.4, kde lze pod nulovou osou spatřit zahazování paketů modré barvy. Tato situace nastává v případě, pokud je služba využívána víc, než je přidělená přenosová rychlost v dané třídě. S pomocí QoS si tato třída může vypůjčit nevyužitou přenosovou rychlost z jiné třídy. Problém je celková přenosová rychlost dovedena na server. Tato přenosová rychlost nemusí stačit pro vyhrazené služby. V takovém případě se začnou zahazovat pakety i z jiných tříd a to tedy i s vyšší prioritou. V této situaci musíme vyřešit vhodný poměr rozdělení přenosové rychlosti mezi jednotlivými službami a nastavit jej tak, aby se nezahazovaly pakety vyšších priorit. Další jedinou a nejvhodnější variantou jak tento problém vyřešit je navýšením garantované přenosové rychlosti navedené do serveru. Jakmile se přivede vyšší přenosová rychlost, stačí poté pouze přenastavit přidělené přenosové rychlosti jednotlivým třídám v optimálním poměru. Poté se situace zlepší a přestanou se zahazovat pakety vyšších priorit.

V konečně fázi jsem se rozhodl zhodnotit vytížení serveru. V potaz jsem vzal jak ranní, dopolední tak i večerní hodiny. K tomuto jsem využil utilitu htop. V dopoledních hodinách při využití celkové přenosové rychlosti bylo vytížení procesoru menší než 5%. V odpoledních hodinách se vytížení procesoru zvýšilo o několik procent, ale nepřekročilo hranici 10%. Ve večerních hodinách, kdy je na server největší nápor dosahoval server zatížení okolo 15%.

7 Závěr

Hlavním cílem diplomové práce bylo popsat problematiku tvarování provozu a kvalitu služby. Uvést příklad z praxe, jaké jsou rozdíly mezi konfigurací kvality služby v operačním systému Debian pro několik jednotlivců nebo pro skupinu lidí.

Primárním aspektem bylo rozložit vhodně přivedenou přenosovou rychlost pro našich víc jak 900 klientských stanic. Hlavní myšlenka byla vytvořit kvalitu služby tak, aby všichni klienti byly obsloužení rovnoměrně. Snahou bylo tedy najít vhodnou frontovou disciplínu dle našich nároků. Pro naše potřeby jsem se rozhodl využít disciplínu HTB, díky které jsem mohl v konečném důsledku optimálně nastavit kvalitu služby. Dále vhodné rozdělení přenosové rychlosti do tříd, které budou tvořit vyhrazenou přenosovou rychlost dle nejpoužívanějších služeb. Jako příklad jsem uvedl v praxi služby, které jsem podle důležitosti řadil do tříd s různou prioritou. V postupu práce jsem uvedl příklad, jak se dá kvalita služby nastavit pro pár jednotlivců nebo i pro skupinu lidí.

V celém návrhu bylo využito řazení jednotlivých služeb do 4 tříd. Tyto třídy tvořili v praxi datové toky označené jako stahovači, ICMP a DNS služby, SIP služby společně s webovými a emailovými službami. Nakonec byla ještě vytvořena lokální služba pro rychlost jednotlivých klientů. Novinkou přibyla i změna v rozpoznání klientů, kteří využívají nedbale přenosovou rychlost a omezují tak přenosovou rychlost ostatních klientů nebo služeb. Pro tuto schopnost jsem zvolil metodu kontrolování toků všech klientů pomocí CONNBYTES v IPTABLES. Tato vlastnost funguje na principu označování klientů s nastavenou délkou nepřerušeného spojení do downloadu. V praxi se toto velmi dobře osvědčilo a pomohlo to ke kvalitnějšímu nastavení kvality služby. Dále v třetím návrhu byly poprvé mapované veřejné IP adresy na IP adresy klientů ve vnitřní síti.

Výstupem celé práce jsou grafy v kapitole vyhodnocení. V této sekci jsou na grafech popsány, jak funguje kvalita služby a poukázání na nejvyužívanější typy komunikace.

Diplomová práce je postavena na zkušebním nasazení serveru v reálných podmínkách. Přivedená je garantovaná přenosová rychlost 1000 Mbit/s na síťové rozhraní serveru.

Přínosem této práce je praktická ukázka jak tvarovat provoz a konfigurovat kvalitu služby pomocí operačního systému Debian. Dále využití polí TOS a DSCP pro potřeby kvality služby. Tato práce je vhodná pro lokální ISP. Ti můžou svým klientům nastavit takovou kvalitu služby, o které pojednává tato práce. Samozřejmě s ohledem na počet klientů v síti, od kterého se bude odvozovat i složení serveru a kvalita hardware pro konkrétní požadavky.

Dalším rozšířením této práce je možnost rozšíření o antidos systém. V síti probíhá velké množství útoků, které je nutné řešit. Při rozšíření o antidosový systém, který by řešil tyto útoky, by ulehčil spravování takovéto sítě. Lze tuto práci rozšířit o možnost traffic shappingu ve směru uploadu. Tato práce je postavená na protokolu IPv4. Z toho důvodu jako rozšíření této práce je možné využit novějšího protokolu IPV6.

Použitá literatura

[1] GIAMBENE, Giovanni. Queuing theory and telecommunications. New York: Springer, 2014. ISBN 978-1461440833.

[2] Linux Advanced Routing & Traffic Control HOWTO. Linux Advanced Routing & Traffic Control HOWTO [online]. Dostupné z: http://lartc.org

[3] NG, Chee Hock. a Boon-Hee. SOONG. Queueing modelling fundamentals with applications in communication networks. 2nd ed. Hoboken, NJ: Wiley, c2008. ISBN 978-0470519578

[4] GHEORGHE, Lucian. Designing and implementing Linux Firewalls and QoS: using netfilter, iproute2, NAT, and L7-filter ; learn how to secure your system and implement QoS using real-world scenarios for networks of all sizes. Birmingham [u.a.]: Packt Publ, 2006. ISBN 9781904811657.

[5] SETH, Sameer a M. AJAYKUMAR VENKATESULU. TCP/IP architecture, design and implementation in Linux. Hoboken, N.J: Wiley, 2008. ISBN 9780470377833.

[6] KRAFFT, Martin F. The Debian system: concepts and techniques. San Francisco: No Starch Press, 2005. ISBN 978-1593270698.

[7] NEMETH, Evi, Garth SNYDER a Trent R. HEIN. Linux: kompletní příručka administrátora. Brno: Computer Press, 2004. Administrace (Computer Press). ISBN 80-722-6919-4.

[8] Advanced traffic control - ArchWiki. ArchWiki [online]. Dostupné z: https://wiki.archlinux.org/index.php/Advanced_traffic_control

[9] Tucny. Tucny [online]. Copyright © 2018 D Tucny. All rights reserved [cit. 28.04.2018]. Dostupné z :https://www.tucny.com/Home/dscp-tos

[10] IPP2P – kladivo na stahovače - Linux E X P R E S. Linux E X P R E S [online]. Copyright © 2018 CCB, spol. s r. o., všechna práva vyhrazena. [cit. 28.04.2018]. Dostupné z: https://www.linuxexpres.cz/praxe/ipp2p-kladivo-na-stahovace

[11] Understanding IP Addressing and CIDR Charts — RIPE Network Coordination Centre. RIPE Network Coordination Centre [online]. Dostupné z: https://www.ripe.net/about-us/press-centre/understanding-ip-addressing

[12] HTB - jemný úvod - Root.cz. Root.cz - informace nejen ze světa Linuxu [online]. Copyright © 1998 [cit. 28.04.2018]. Dostupné z: https://www.root.cz/clanky/htb-jemny-uvod/

[13] How can I do traffic shaping in Linux by IP? - Server Fault. Server Fault [online]. Dostupné z: https://serverfault.com/questions/191560/how-can-i-do-traffic-shaping-in-linux-by-ip

[14] Create Policies for QoS on your Debian Linux System. Open Standards Software IP PBX - 3CX [online]. Dostupné z: https://www.3cx.com/blog/voip-howto/qos-linux/

[15] qos - Default mark for packets using iptables - Unix & Linux Stack Exchange. Unix & Linux Stack Exchange [online]. Dostupné z: https://unix.stackexchange.com/questions/2259/default-mark-for-packets-using-iptables

[16] Debian advanced router for ISP - firewall, traffic shaping, smp_affinity, taskset, sysctl and more ... - АЙТИСЪРВИС 2009 ЕООД. Айтисървис - Интернет, хостинг, колокация, мониторинг [online]. Copyright ©2009 [cit. 28.04.2018]. Dostupné z: https://itservice-bg.net/debian-advanced-router/

[17] TrafficControl - Debian Wiki. FrontPage - Debian Wiki [online]. Dostupné z: https://wiki.debian.org/TrafficControl

[18] Traffic Control using tcng and HTB HOWTO. http://linux-ip.net/ [online]. Copyright © 2006, Martin A. Brown [cit. 28.04.2018]. Dostupné z: http://linux-ip.net/articles/Traffic-Control-tcng-HTB-HOWTO.html

[19] Traffic Shaping, Bandwidth Shaping, Packet Shaping with Linux tc htb. IP Location Finder - Geolocation [online]. Copyright © 2006 [cit. 28.04.2018]. Dostupné z: https://www.iplocation.net/traffic-control

[20] [online]. Dostupné z: http://tldp.org/HOWTO/Adv-Routing-HOWTO/lartc.qdisc.filters.html

[21] Linux TC (帯域制御、帯域保証) 設定ガイドライン | GREE Engineers' Blog. 301 Moved Permanently [online]. Copyright © 2018 GREE Engineers [cit. 28.04.2018]. Dostupné z: http://labs.gree.jp/blog/2014/10/11288/

[22] HTB home. [online]. Dostupné z: http://luxik.cdi.cz/~devik/qos/htb/

[23] Využívání vymezených rádiových kmitočtů | Český telekomunikační úřad. Český telekomunikační úřad [online]. Copyright © 2018 [cit. 02.07.2018]. Dostupné z: https://www.ctu.cz/vyuzivani-vymezenych-radiovych-kmitoctu

Seznam příloh

Součástí DP je CD.

Adresářová struktura přiloženého CD: firewall.sh, qos.py, KRM0009-DP.pdf

Příloha A: *Naměřené hodnoty*

Tabulka A.1: *Větší tabulka naměřených hodnot*

Tabulka A.2: *Jiná tabulka*

Příloha B: *Velká tabulka na celou stránku*

text							

Zdrojový kód firewall.sh

```
#!/bin/bash

# konektivita  eth1
# vnitrni sit RDLnet eth2
# Rozsah addres 10.0.0.0/8, 192.168.0.0/12, 172.16.0.0/16, 85.207.92.16/28

echo "Nastavovani firewall"
# Vyčištění předchozích nastavení firewall, nat a mangle
iptables -F
iptables -X
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -F -t nat
iptables -F -t mangle
# Povolení pro průchod mezi dvěma ethernety (mezi eth1 a eth2) a všecho co přijde z lo
iptables -A INPUT -i lo -j ACCEPT
iptables -A FORWARD -i eth1 -o eth2 -j ACCEPT
iptables -A FORWARD -i eth2 -o eth1 -m state --state ESTABLISHED,RELATED -j ACCEPT
# Příjmuté ICMP a SSH   a TCP/UDP rovnou propouštíme ven ze sítě
iptables -A INPUT -p icmp -j ACCEPT #ICMP
iptables -A INPUT -p tcp --dport 22 -j ACCEPT #SSH
iptables -A INPUT -p tcp --dport 2223 -j ACCEPT #TCP/UDP
# Pro přístup VPN, webu a portu Mikrotik
iptables -A INPUT -p udp --dport 1194 -j ACCEPT
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp --dport 8080 -j ACCEPT
```

```
# Průchod pro pakety DNS TCP/UDP
iptables -A INPUT -p tcp --dport 53 -j ACCEPT
iptables -A INPUT -p udp --dport 53 -j ACCEPT
# Nastavení NAT pro propouštění paketů z vnitřní sítě 10.0.0.0/8
na výstupní WAN adresu 85.207.92.18/28
iptables -t nat -A POSTROUTING -o eth0 -s 10.0.0.0/8 -j SNAT --
to-source 85.207.92.16/28
# Routované veřejné IP
iptables -A FORWARD -o eth3 -s 85.207.92.16/26 -j ACCEPT
# Mapovani verejnych adres na klientské interní adresy
iptables -t nat -I PREROUTING -d 85.207.92.19 -i eth3 -j DNAT --
to 10.120.17.5
iptables -t nat -I PREROUTING -d 85.207.92.20 -i eth3 -j DNAT --
to 10.120.18.8
iptables -t nat -I PREROUTING -d 85.207.92.21 -i eth3 -j DNAT --
to 10.120.19.14
iptables -t nat -I PREROUTING -d 85.207.92.22 -i eth3 -j DNAT --
to 10.120.20.17



# Část pro QoS
# Vytvoření pravidla pro frontu mark_stahovac, první si
předchozí nastavení vymažeme pomocí prvních 2 řádků, třetí řádek
vytvoříme novou frontu.
iptables -t mangle -F mark_stahovac
iptables -t mangle -X mark_stahovac
iptables -t mangle -N mark_stahovac
# Zkontrolujeme jestli existuje nějaká předchozí marka
iptables -t mangle -A PREROUTING -j CONNMARK --restore-mark
```

```
#Nastavíme marku pro stahovace

iptables -t mangle -A PREROUTING -m mark --mark 0 -j
mark_stahovac

#Přiřadíme k mark pro stahovače hodnotu v TOS v DSCP

iptables -t mangle -A mark_stahovac -m dscp --dscp 0x02 -j MARK
--set-mark 0x08

# Marku následovně uložíme

iptables -t mangle -A mark_stahovac -j CONNMARK --save-mark


# Monitorujeme dlouhé spojeni downloadu kvůli rozlišení klienta
stahovače od ostatních klientů

iptables –t mangle –A FORWARD –m connbytes –p tcp –conbytes
600000 –connbytes-dir –connbytes-mode bytes –j mark_stahovac

#Řazení DNS,sip,http a email

iptables -t mangle -A FORWARD -p udp --sport 53 -m dscp --dscp
0x00 -j DSCP --set-dscp 0x00

iptables -t mangle -A FORWARD -p udp -m multiport -dports
5060,80,443,143,993,995,587 -m dscp --dscp 0x01 -j DSCP --set-
dscp 0x01



 Povolíme ip_forward pro náš průchod mezi dvěma interface eth1 a
eth2

echo 1 > /proc/sys/net/ipv4/ip_forward
```

Zdrojový kód qos.py

```
#!/usr/bin/python
dev_inet=eth2 #LAN
dev_onet=eth1 #WAN
#Vymezení přenosových rychlostí
bw = 100000
bw_public = 100000
bw_fast = 55000
bw_web = 20000
bw_other = 10000
bw_stahovac = 10000
# Monitorovaní download
# Vytvoření root interface na eth2 a následovně vytvoříme třídu
tc qdisc add dev %(dev_inet)s root handle 1: htb default 20
tc class add dev %(dev_inet)s parent 1: classid 1:1 htb burst
10000 cburst 10000 rate %(bw)kbit ceil %(bw)kbit
tc class add dev %(dev_inet)s parent 1:1 classid 1:2 htb burst
10000 cburst 10000 rate %(bw_public)kbit ceil %(bw_public)kbit
prio 1
# fronta pro SIP,SMTP,HTTP,HTTPS,POP3,IMAP
tc class add dev $dev_inet parent 1:0 classid 1:10 htb burst
10000 cburst 10000 rate $(( $bw_fast ))kbit ceil $(( $bw_fast
))kbit prio 0

#rychlá fronta pro DNS a ICMP
tc class add dev #dev_inet parent 1:0 classid 1:11 htb burst
10000 cburst 10000 rate $(( $bw_web ))kbit ceil $(( $bw_web
))kbit prio 1
#Fronta pro ostatní provoz
tc class add dev #dev_inet parent 1:0 classid 1:12 htb burst
10000 cburst 10000 rate $(( $bw_other ))kbit ceil $(( $bw_other
))kbit prio 1
```

```
#Fronta pro stahovače

tc class add dev $dev_inet parent 1:0 classid 1:13 htb burst
10000 cburst 10000 rate $(( $bw_stahovac ))kbit ceil $((
$bw_stahovac ))kbit prio 1

# Nastavení frontových disciplín

tc qdisc add dev $dev_inet parent 1:10 handle 10:0 sfq perturb
10

tc qdisc add dev $dev_inet parent 1:11 handle 11:0 sfq perturb
10

tc qdisc add dev $dev_inet parent 1:12 handle 12:0 sfq perturb
10

tc qdisc add dev $dev_inet parent 1:13 handle 13:0 sfq perturb
10

# Přidáme filtry pro jednotlivé třídy

tc filter add dev $(dev_inet) protocol ip parent 1: pref 10 u32
match u8 0x00 0xfc at 1 flowid 1:10

tc filter add dev $(dev_inet)s protocol  ip parent 1: pref 10
u32 match u8 0x04 0xfc at 1 flowid 1:11

tc filter add dev $(dev_inet)s protocol  ip parent 1: pref 10
u32 match u8 0x08 0xfc at 1 flowid 1:13


# Monitorovaní upload

# Vytvoření root interface na eth1 a následovně vytvoříme třídu

tc qdisc add dev %(dev_onet)s root handle 1: htb default 20

tc class add dev %(dev_onet)s parent 1: classid 1:1 htb burst
10000 cburst 10000 rate %(bw)kbit ceil %(bw)kbit

tc class add dev %(dev_onet)s parent 1:1 classid 1:2 htb burst
10000 cburst 10000 rate %(bw_public)kbit ceil %(bw_public)kbit
prio 1

#rychlá fronta ICMP, DNS

tc class add dev $dev_onet parent 1:0 classid 1:10 htb burst
10000 cburst 10000 rate $(( $bw_fast ))kbit ceil $(( $bw_fast
))kbit prio 0
```

```
#fronta pro SIP,SMTP,HTTP,HTTPS,POP3,IMAP

tc class add dev #dev_onet parent 1:0 classid 1:11 htb burst
10000 cburst 10000 rate $(( $bw_web ))kbit ceil $(( $bw_web
))kbit prio 1

#Fronta pro ostatní provoz

tc class add dev #dev_onet parent 1:0 classid 1:12 htb burst
10000 cburst 10000 rate $(( $bw_other ))kbit ceil $(( $bw_other
))kbit prio 1

#Fronta pro stahovače

tc class add dev $dev_onet parent 1:0 classid 1:13 htb burst
10000 cburst 10000 rate $(( $bw_stahovac ))kbit ceil $((
$bw_stahovac ))kbit prio 1

# Nastavení frontových disciplín

tc qdisc add dev $dev_onet parent 1:10 handle 10:0 sfq perturb
10

tc qdisc add dev $dev_onet parent 1:11 handle 11:0 sfq perturb
10

tc qdisc add dev $dev_onet parent 1:12 handle 12:0 sfq perturb
10

tc qdisc add dev $dev_onet parent 1:13 handle 13:0 sfq perturb
10
```